ABSTRACT

Title of thesis: LARGE MOTION VISUALIZATION AND ESTIMATION FOR FLAPPING WING SYSTEMS

Tzu-Sheng Hsu, Master of Science, 2015

Thesis directed by: Professor Balakumar Balachandran
Department of Mechanical Engineering

Studies of fluid-structure interactions associated with flexible structures such as flapping wings require the capture and quantification of large motions of bodies that may be opaque. Motion capture of a free flying insect is considered by using three synchronized high-speed cameras. A solid finite element representation is used as a reference body and successive snapshots in time of the displacement fields are reconstructed via an optimization procedure. An objective function is formulated, and various shape difference definitions are considered.

The proposed methodology is first studied for a synthetic case of a flexible cantilever structure undergoing large deformations, and then applied to a Manduca Sexta (hawkmoth) in free flight. The three-dimensional motions of this flapping system are reconstructed from image date collected by using three cameras. The complete deformation geometry of this system is analyzed. Finally, a computational investigation is carried out to understand the flow physics and aerodynamic performance by prescribing the body and wing motions in a fluid-body code.

This thesis work contains one of the first set of such motion visualization

and deformation analyses carried out for a hawkmoth in free flight. The tools and procedures used in this work are widely applicable to the studies of other flying animals with flexible wings as well as synthetic systems with flexible body elements.

LARGE MOTION VISUALIZATION AND ESTIMATION FOR FLAPPING WING SYSTEMS

by

Tzu-Sheng Hsu

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Science
2015

Advisory Committee:
Professor Balakumar Balachandran, Chair and Advisor, Mechanical Engineering
Professor Peter Bernard, Department of Mechanical Engineering
Associate Professor Peter W. Chung, Department of Mechanical Engineering

For all the unforgettable moments in Maryland.

Acknowledgments

I would like to thank my advisor, Professor Balachandran, for his guidance and overwhelming trust to bring out the best in me. This mentorship began with an email, while I was still in Taiwan and was a stranger with an undergraduate degree across the Pacific ocean. I would also like to thank my committee members, Professor Chung and Professor Bernard, for their assistance in completing this thesis.

I would like to thank Tim Fitzgerald for his mentoring with my research and his time spent teaching me how to do research by every email, debug, and discussion in weekdays, weekends, days, and nights. I would like to thank Vincent Phuc Nguyen for his involvement in the formulation of the core principles and every bottleneck associated with this work. Every hours-long discussion on Friday afternoon has made me become a better thinker.

I am grateful to all my lab mates whom I feel fortunate to call my friends. I would like to list the names in order of the time I knew them, which would be Nicholas Vlajic, Edmon Perkins, Hesham Ismail, Celeste Poley, Wenhua Hu, Marcelo Valdez, Chris Chabalko, Ayan Moitra, Bryan Hays, Nick Wondolowski, Telly Saliou, Ryan Miller, Abdullah Zibdeh, Vipin Agarwal, Sergio Preidikman, and Trisha Patel. It couldn't have been an enjoyable and insightful experience without your company.

I am thankful for NSF for funding this work, with Grant No. CMMI-1250187. At last, I would like to thank my family in the west coast and in Taiwan for their support. The bonding of the Hsu's is the thing we are all proud of.

Table of Contents

List of Tables

List of Figures

Nomenclature

FSI	Fluid-structure interactions
FFT	Fast Fourier transform or discrete Fourier transform
FE	Finite element
UVLM	Unsteady vortex lattice method

Chapter 1: Introduction

1.1 Problem of interest

The mechanisms underlying flapping flight have drawn extensive attention from various research communities over the last several decades. Larger species of flapping insects, such as the hawkmoth and dragonfly, as well as small birds, such as the hummingbird, lie within the range of interest for micro-air-vehicle (MAV) design. For this reason, studying the mechanism of how these animals fly has become a research topic of much interest. Currently, the understanding of flapping wing mechanism is still under development. Researchers are experimentally studying the state-of-art of flapping mechanisms and associated fluid-structure interactions between the wing and the fluid, along with modelling and simulations of the motions. In addition, a high-resolution measurement of the deformation geometry flexible wing such as a hawkmoth wing is unavailable at this time.

An objective of this work is the development of a framework to estimate the deformation geometry of a flapping wing, as it is being used in flight. The flapping motion is first reconstructed, and then this reconstructed observation is analyzed. Furthermore, by prescribing the flapping motion into a fluid-body model, the aerodynamic performance has also been numerically studied.

1.2 Prior work

In experiments used for capturing the motions of flapping creatures, multiple cameras are commonly used to observe the kinematics of flight. Generally speaking, two approaches have been followed, namely, feature tracking and three-dimensional (3D) reconstruction[3]. 3D direct reconstruction provides a point cloud representing the intersection of all the images extruded in the third direction. It is limited by volumetric accuracy, which decreases as the characteristic length of the target object and the magnitude of deformation increase.

On the other hand, one identifies specified landmarks on the target object and this tracking data is further used to analyze the kinematics of the motion. Ortega-Jimenez, Greeter, Mittal, and Hedrick [4] have inferred wing motions by identifying the locations of wing tips. Dong [5] and Koehler, Liang, Gaston, Wan, and Dong [6] have suggested the use of artificial marks on wings, to observe wing topology and deformation. With such a technique, one is able to reconstruct two-dimensional (2D) wing surfaces. By using the tracked landmarks, the wing can be meshed evenly into grids in each frame independently[7]. However, the marking process can be time consuming. In addition, the number of marker points are inevitably limited. If the target object is opaque, the feature points can be hardly seen from all angles simultaneously. Further information can be found in reference [3].

For the purpose of deducing flight characteristics, researchers often use the following parameters and means to characterize the flight motion:

(1) General rigid body motion using Euler angles [8, 9] or quaternions [10].

By examining the time histories of the angles, the amplitudes and phase relations between different points on the wing can be explored. (2) Mean stroke angle [8]. (3) Wing tip trajectories [4, 8]. (4) Forward flight speed. The target subject is placed in a wind tunnel to observe changes in the previously mentioned characteristics with respect to the wind speed [8, 11]. (5) Camber angle describing the deformation [6].

Aerodynamic performance of the recorded insect flight can be estimated numerically given the measured motion[12, 13, 14]. Features and aspects such as the vorticity structures [15], wing-wake interaction, wing-wing interaction, and so on [16] have been reported; these features and aspects have also been reported in experimental studies of insect flight and dynamically scaled robotic wings [17, 18].

In order to gain a further understanding of the flight mechanisms and explain why insects fly in the observed manner, fluid-structure interaction(FSI) simulations can play an important role for assessing the structural information of wings and help implement a more comprehensive study, with additional control variables such as beating frequency, stiffness of wing, and so on. From a simplified simulation environment, 2D studies can provide insights into the effects of elasticity of the wing [19, 20], the clap and fling mechanism[21], and the wing-wake interaction [7]. General 3D simulation environments have been developed more recently [22, 23].

Most FSI studies are still limited by the lack of full deformation geometry in insect flight, which can serve as a reference. Moreover, in most of the studies, one uses a linear material model to construct the structure of the wing, while the deformation is underestimated.

Nonlinear phenomena are often considered as well for the reason that the flap-

ping frequency is much lower than the linear natural frequency of the wing. Vanella et al. [19] suggest that a nonlinear resonance has been invoked, while Ramananarivo, Godoy-Diana, and Thiria [24] argue that the flapping frequency favors the phase lag from leading edge to trailing edge and improves aerodynamic efficiency, as a consequence.

1.3 Objectives and approach

As a continuation of the previous work at the University of Maryland, the author endeavors to develop a tool to estimate the wing deformation configuration from multiple cameras for the use of integrating experimental data into fluid-structure simulations. A hyperelastic material finite element model is utilized in order to capture the large deformation of the flapping wings of *Manduca sexta* during free-flight motion. Further investigations into the captured motions have been performed. The research approach for this thesis is shown in Figure 1.1.

1.4 Outline

The rest of the thesis is organized in the following manner: In Chapter 2, the experimental setup is depicted in detail. Preliminary steps necessary to be taken for the use of later chapters are shown, including image processing, camera calibration, and image segmentation. Since, in this step, one uses a reduced-order model to describe the motion of the hawkmoth, some characteristics of the flight motion are discussed. It is mentioned that the gray-banded region in all time histories plots

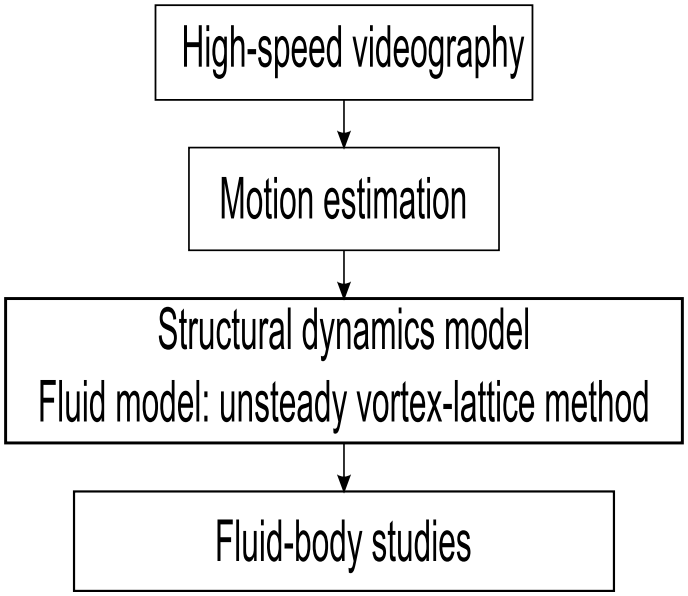


Figure 1.1: Flow chart of the research approach after motion reconstruction for hawkmoth from high-speed videography.

denotes the approximated instances of the downstrokes. In Chapter 3, the author proposes a template-based approach to estimate the deformation geometry of the moth wing. As a case study, the algorithm is first tested with a synthetic flexible structure undergoing large motions, before this method is applied to the image data of the free flight motion of *Manduca sexta*. The fluid-body analysis is presented in Chapter 4. Finally, the summary and some suggestions for future work are provided in Chapter 5. Appendices that provide some background knowledge for this thesis and references are included at the end.

Chapter 2: Experimental arrangement and motion visualization

In this chapter, the author describes the experimental arrangement used to capture the free flight motion of *Manduca sexta* in detail. The intermediate steps taken to make the raw data applicable to the methods presented in the latter chapters are also demonstrated.

2.1 Preparation of the objects to be filmed

The subject chosen to be filmed is the *Manduca sexta*, which is also known as the hawkmoth. The pupae were obtained from Carolina Biology Supply (*http://www.carolina.com/*) and contained in the mesh cage. One visible light source and two heat lamps were used to provide adequate light and heating. The temperature was held between 27 to 30 degree Celsius by adjusting the height of the lamps. At least 8 hours of light was found to be needed to be provided per day, otherwise, the pupae may go into diapause. A humidifier was used to maintain the humidity level between 35% to 40%, and some tree branches were provided for the moths to climb up and dry their wings. From the research group's past experience, the moths emerge slightly before dawn after 2 to 3.5 weeks after the process was initiated.

After their emergence, it takes approximately one day for the moths to learn

Figure 2.1: Environment where the moths were raised. The mesh cage is put into a box to contain more heat, and the cage was surrounded by lamps. A water tank was used to maintain a specified range of humidity. A thermometer can be located at the bottom of the image.

how to fly. Putting them into a bigger space greatly expedites the learning process. The author and co-workers also attached window screens to the walls and floor of the observation chamber to help them climb up and crawl. However, their wings were found to be impaired easily due to the limited space. A mature moth can live between 3 to 5 days. They are found to be most active in the first few days before laying eggs.

2.2 Videography

The focus of this work has been on motion reconstruction from the outline of the images without utilizing information from feature tracking. A schematic plot of the camera positions is shown in Figure 2.2. As shown in Figure 2.2, three Phantom Miro-M310 cameras(Vision Research, Wayne, NJ, USA) are used to provide orthogonal views at 2000 frames per second with a resolution of 1280 times 800 pixels. The video subjects (hawkmoths) were allowed to fly freely with the beating frequency of approximately 25 Hz in a $2ft \times 2ft \times 2ft$ plexiglass flight chamber. Each

video was filmed with an exposure time of 200 μs and aperture value $f/4$, while the three cameras form a focal box. The cameras were synchronized through F-sync and connected to an external event trigger. Two cameras were set as the external clocks to receive the time information from the other camera, which is set as the master clock. Two break beam laser sensors, Banner QS30LLPC, are connected to a circuit board, which were used to send high and low value signals to all the external trigger ports at the cameras simultaneously. The laser beams intersect at center of the focal box.

The cameras were set to record continuously. The cameras each have an circular buffer. This permits post triggering of an event. When a subject breaks both lasers, several seconds prior to the break would be saved and then transferred to an external computer hard drive. A picture of the experimental arrangement is shown in Figure 2.3

In order to film the flight of nocturnal insect like *Manduca sexta*, infrared LED lights were used to provide backlit illumination, which provided sharp outlines of the target. The wavelength of the light source, 650 nm, is below the visible region of the moth. Note that the spectral response of each camera extends to a range higher than the visible light wavelength range.

2.3 Image processing

The goal of image processing is to provide clear positions of the outlines of the shadow. Such procedure provides fast and robust location information of the

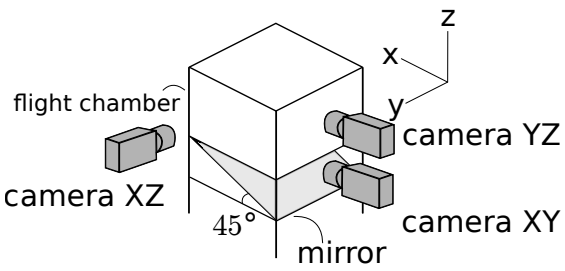


Figure 2.2: Camera arrangement.

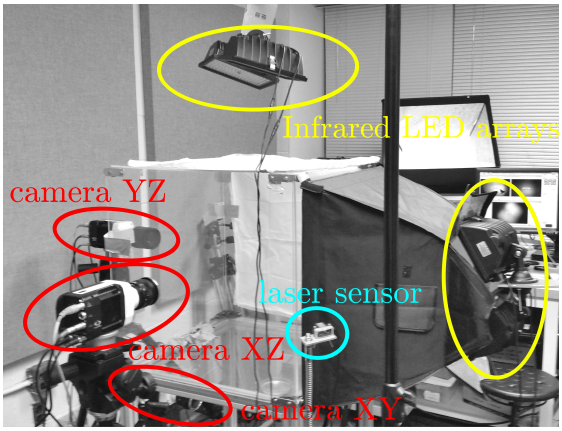


Figure 2.3: Experimental arrangement.

boundaries of the target object. Here, the author started with the Matlab code provided on the website of the Itai Cohen Group [25] and customized it for the specific scenario in the current research. The author further extracted the boundary points from the shadow using a simple cubic marching scheme, as shown in Figure 2.5. By comparing every four neighboring pixels in the image, the boundaries were identified for which the four pixel intensities dramatically changed. Note that curve fitting schemes were not applied on the boundaries as done in most literature reported in image processing. The idea of curve fitting is embedded in the motion reconstruction algorithm, which will be shown in a later chapter.

2.4 Camera projections

For some different types of the sensors, the projections are not as simple as orthogonal projections. In our case, high-speed cameras are used to capture the motion of free-flight of *Manduca sexta*, and due to the high insect size to filming distance ratio, the perspective distortion is not negligible.

PHANTOM V21X / V31X / V41X / V61X / V12.1 / V71X / Miro310 / Miro110

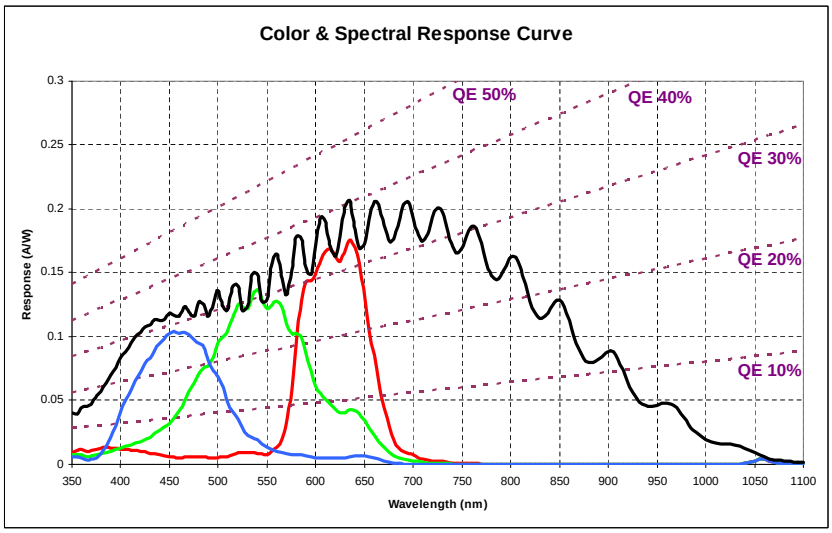


Figure 2.4: Wave spectral response of the Phantom Miro-M310.
(http://www.visionresearch.com/Products/High-Speed-Cameras/Miro-M310/)

A camera is commonly modelled as a pin-hole camera [2]. The image sensor that can be used to capture a point in the R' frame is located at the intersection of the line between the point to the focal point and the image sensor plane.

As shown in Figure 2.7, the relation between a point in the world frame (X, Y, Z) and the corresponding image point (x, y) is

$$\begin{aligned} x &= f\frac{X}{Z} \\ y &= f\frac{Y}{Z} \end{aligned} \tag{2.1}$$

or equivalently,

$$
\begin{aligned}
w \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} &= \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \\
&= \mathbf{K} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}
\end{aligned} \tag{2.2}
$$

Here, the origin is the focal point C, and f represents the focal length. w is an arbitrary factor representing the colinearity characteristic of the equation. The matrix $\mathbf{K}$ is often known as the intrinsic matrix, which consists of the intrinsic parameters of the camera.

Now, consider a more general case of the intrinsic matrix:

$$
\mathbf{K} = \begin{bmatrix} \alpha_x f & s & x_0 \\ 0 & \alpha_y f & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.3}
$$

Here, the α values represent the size differences of the length and width of the pixel sensor. Ideally, the pixel sensor is square if one uses a monochrome camera. s is the skew factor, which in most of the cases, can be neglected. (x_0, y_0) is the principal point. The principal point is the projection of the focal point on the image plane along the Z-direction, or the principal direction. Note that for most of the image processing toolboxes, the origin of the image is at the upper-left corner.

In practice, one may define the transformation between the R'-frame and the coordinate aligned to the principal direction. This in mind, equation (2.2) is gener-

alized to

$$
\begin{aligned}
w \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} &= \mathbf{K} \left[\mathbf{R} \mid \mathbf{t} \right] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \\
&= \mathbf{P} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}
\end{aligned}
\tag{2.4}
$$

$\mathbf{R}$: rotation matrix

$\mathbf{t}$: translation vector

$\mathbf{P}$: camera matrix

Essentially, there are 11 variables: 5 intrinsic parameters and 6 degrees of freedom associated with rigid-body motion. The details of the steps to find the parameters are provided in Section A.2

2.5 Image segmentation

2.5.1 Body tracking

Before estimating the wing motion, identification of the body part of the hawkmoth is necessary. An image consists of wings and body blended together due to the occlusion in 3D. Here, the author identifies the locations of the pixels that belong to the body. Images containing only the body part of the hawkmoth from one time frame were obtained by manually cropping the raw images. Next, direct

reconstruction of the body images was implemented by brute force search in 3D. To be pragmatic, considering the compromise between accuracy and compactness of the data size, an interval of 0.3 millimeters of search was chosen. A point in 3D space is inside the body if its projections exist on all images. Note that the idea is identical to the silhouette cone intersection method discussed in a later section. The result of the point cloud is shown in Figure 2.8. The reference body cloud is then used to fit over the time frames by assuming rigid-body motion. For each time frame, the optimization procedure is executed, with the cost function to be minimized constructed as the union of area between the captured and the projected images from the reference body cloud.

The results of optimization problem at selected frames are shown in Figure 2.6. Note that the proposed method is applicable to any set of data obtained by using multiple cameras. Due to the limited amount of time, however, the results of the remaining work are all applied on the same set of data. The trajectory of the moth body center and time series of the orientation are shown in Figure 2.8, Figure 2.9, Figure 2.11, and Figure 2.12, respectively. The general motion of the body is diagonally upward, while the later half of the motion in the y-direction changes.

2.5.2 Customized reference FE model generation

A reference FE of the moth wing is built by using an open source mesh generation software Gmsh [26] to fit the image data. Although the dimensions of the hawkmoth wings are quite similar between each other, having a customized refer-

ence model for each video subject is still advantageous for motion estimation. Since manual measurements of the dimensions of each subject can be time consuming, a rough estimation directly from the video of flight motion was obtained by modeling the wing as a flat planform extruded in the thickness direction. One reference frame, in which the wing is the closed to the undeformed configuration, is manually chosen. Then, the maximum volume representation of point cloud is obtained by direct reconstruction from images. Since the point cloud that describes the true wing geometry is close to a planar subset of the maximum volume representation, the plane whose intersection to the point cloud has the minimum projection difference is the one that is closet to the true geometry.

By taking some of the contour points and interpolations amongst them by using B-spline, the geometry of the 2D wing planform is determined and then extruded to the third dimension. The author used 27-noded quadratic hexahedron element with the hyperelastic Biot material developed in Fitzgerald's work [22]. This material model is similar to the well-known Kirchhoff model and allows for large rigid body motion and moderate strains.

The result of the customized reference wing is shown in Figure 2.10. There are 51 elements with 726 nodes, associated with each there are three degrees of freedom. The Young's modulus is assumed to be homogeneous across the body.

2.5.3 Leading edge tracking

From observation, the wing motion is found to be actuated at the root position. The wing leading edge is relatively rigid and responds to the actuation instantaneously, with the rest of the wing deforming and following the leading edge motion with a phase difference. From the customized wing obtained from Subsection 2.5.2, the geometry of the leading edge can be used to track through frames. The goal is to fit the reference leading edge points to the corresponding projection images by rigid-body motion. The image points to be fit are the boundary points that describe the contour of the hawkmoth. Since the reference leading edge is fitted to the image points of the boundary points of the hawkmoth, the author defined the cost function as the sum of the closest distances of the projection of each reference leading edge point to the image boundary points. Considering the occlusion of the moth body, if the projection lies in the region where the body is, the cost function was set up to not penalize such a circumstance. Therefore, the image points that describe the hawkmoth body are included with the boundary points. In order to speed up the computations, distance maps were created in advance. The distance map is the same size as the image that each pixel location was used to record the closet distance to the boundary points. Bi-linear interpolation is implemented if the projection lies between pixels. An example of distance map from one image is shown in Figure 2.18. Some snapshots of the estimated leading edge position on the images are shown in Figure 2.13.

Selective frames of the results of the estimated leading edge position are shown

in Figure 2.13. The time histories of the Euler angles in the B-frame are plotted in Figure 2.14 and Figure 2.15. As defined in Figure A.2, the three Euler angles (ρ, θ, ϕ) correspond to the pitch angle, azimuth angle, and the elevation angle relative to B-frame, respectively.

It is observed from the time histories of the elevation angle that the signal at the reversal of the downstroke is flat, and the reversal of the upstroke is sharp, which indicates that the reversal mechanism is much slower at the downstroke than the upstroke. The plateau may be due to the torsional motion during the reversal process, which manifests in the deformation estimation. From the fast-Fourier transform(FFT) analysis, the flapping frequency is estimated as 26.28 Hz. A higher harmonic frequency at 55.19 Hz is also observed.

The phase portraits of one Euler angle against another are plotted in Figure 2.21. Since the signal is noisy, the time derivatives of the Euler angles cannot be accurately estimated. Instead, a plot pseudo-state space is constructed by using delayed coordinate. Assume one is only interested in the two-dimensional state space, displacement $x(t)$ and its time derivative, where the state $x(t)$ can be $\rho(t)$, $\phi(t)$, $\theta(t)$. The delay-coordinate vector y is

$$y = \{x(t) \mid x(t+\tau)\} \tag{2.5}$$

where τ is time delay that needs to be determined. Here, the author has used autocorrelation function to determine the time delay. The autocorrelation function

$C(\tau)$ is given by

$$C(\tau) = \frac{\sum_{k=1}^{N} [x(t_0 + k\tau_s + \tau) - x_{av}] [x(t_0 + k\tau_s) - x_{av}]}{\sum_{k=1}^{N} [x(t_0 + k\tau_s) - x_{av}]^2} \tag{2.6}$$

where

N : number of sampling points.

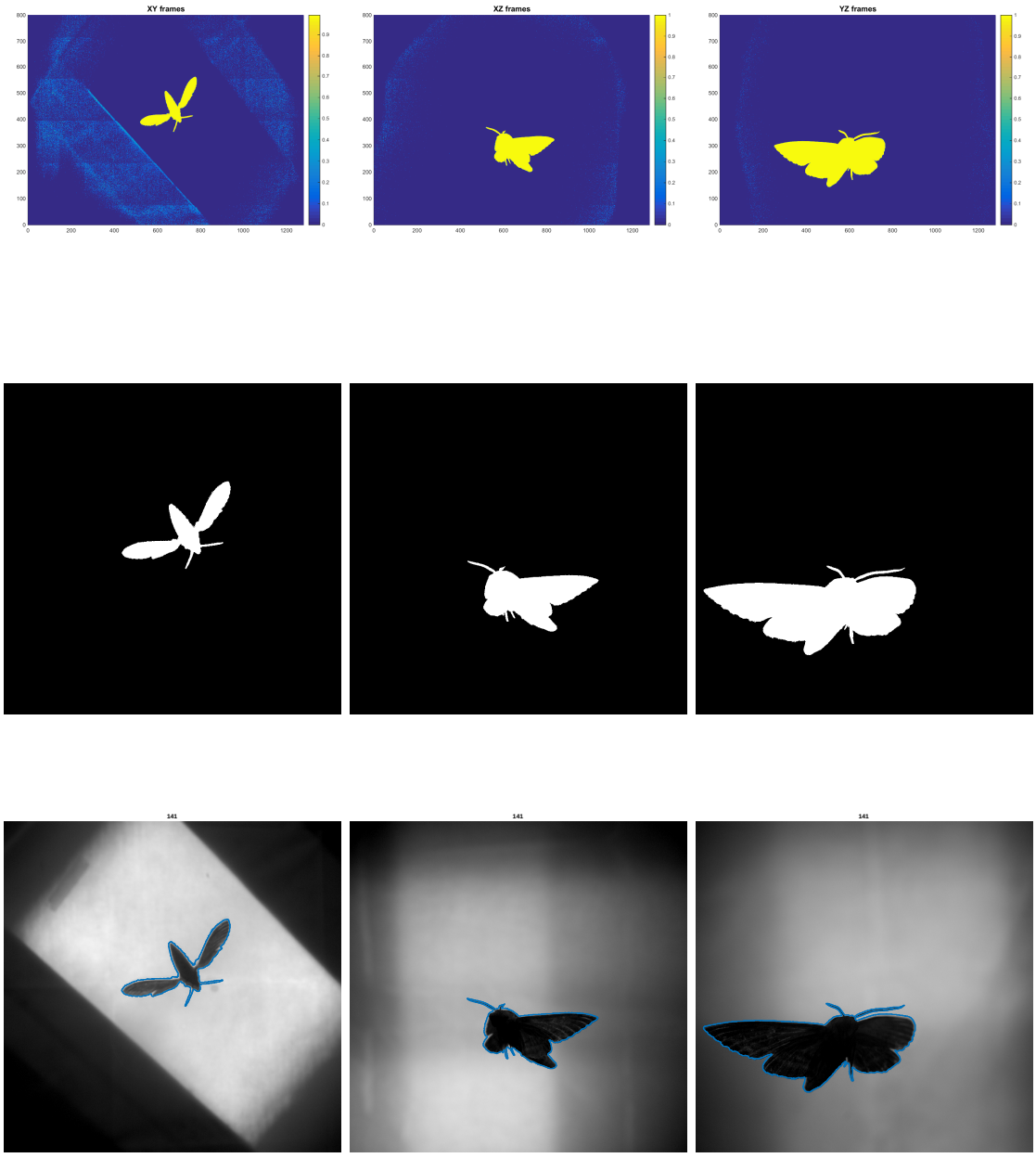
t_0 : start time.

τ_s : sampling time.

x_{av} : average of the signal.

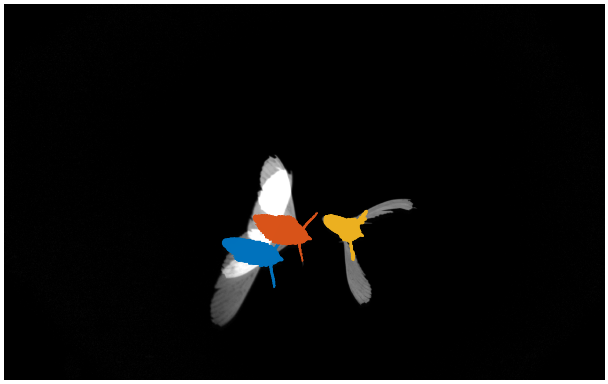
The goal is to find the zero-crossing point of $C(\tau)$ at τ or a minimum value of the correlation function. If a zero crossing does not exist, a local minimum is used to determine the time delay. By choosing τ to be the first zero of $C(\tau)$, one makes $x(t)$ and $x(t + \tau)$ linearly independent[27].

The autocorrelation function for each Euler angle of each wing is shown in Figure 2.19. The time step that is closest to the first zero-crossing point is at approximately a quarter of the wing beat cycle. After using this value as the time delay, one can plot the phase portrait in the reconstructed space, as shown in Figure 2.22.



(a) View from camera XY (b) View from camera XZ (c) View from camera YZ

Figure 2.5: Snapshots of intensity field, binarized image, and edge detection.



(a) View from camera XY

(b) View from camera XZ

(c) View from camera YZ

Figure 2.6: Representative frames of rigid-body motion tracking of the body.

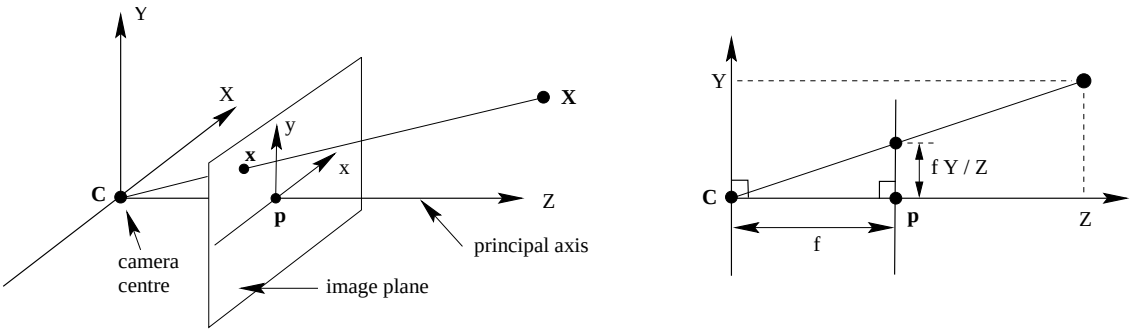


Figure 2.7: Figure of pin-hole camera model [2].

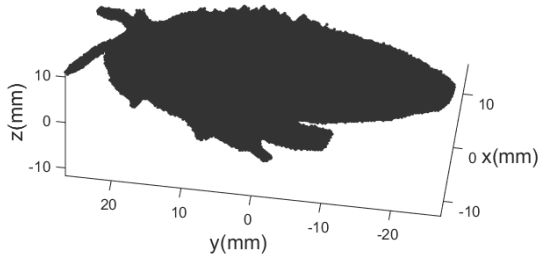


Figure 2.8: Three dimensional direct reconstruction from the body images.

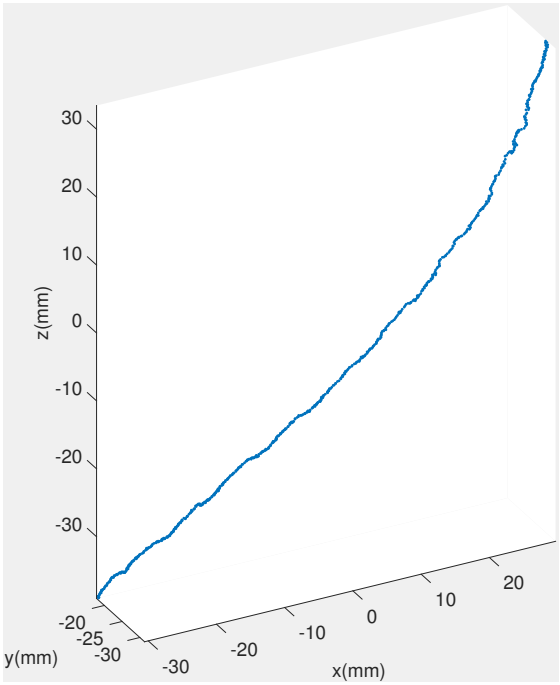


Figure 2.9: Trajectory of the geometric center of the moth body.

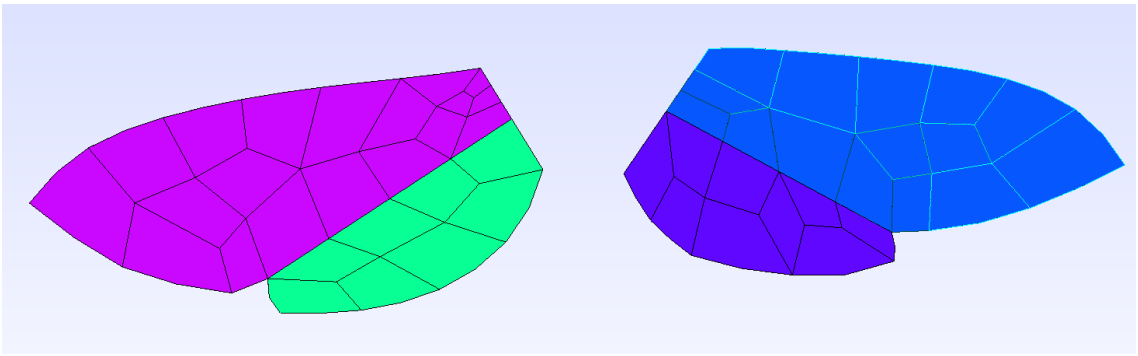
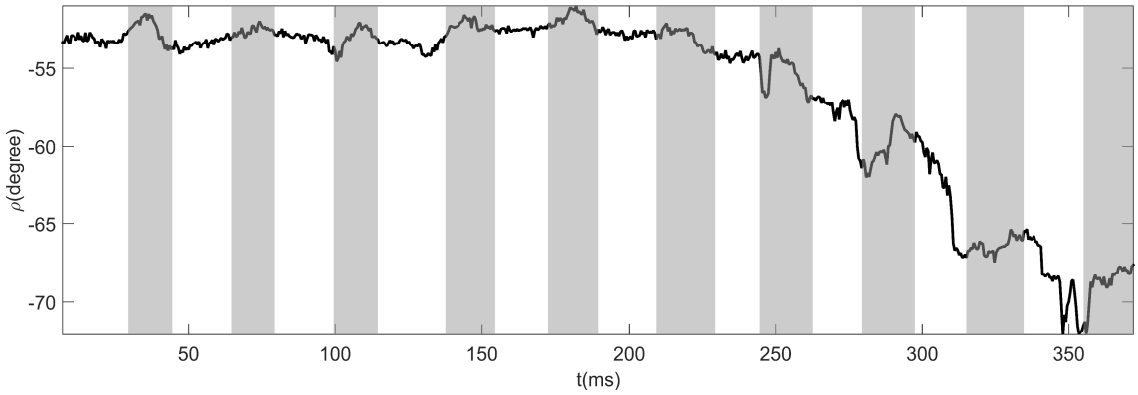
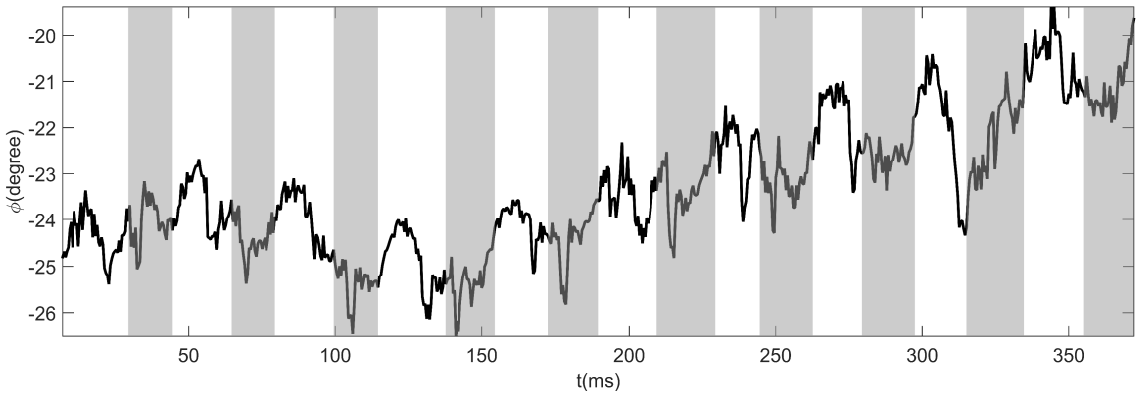


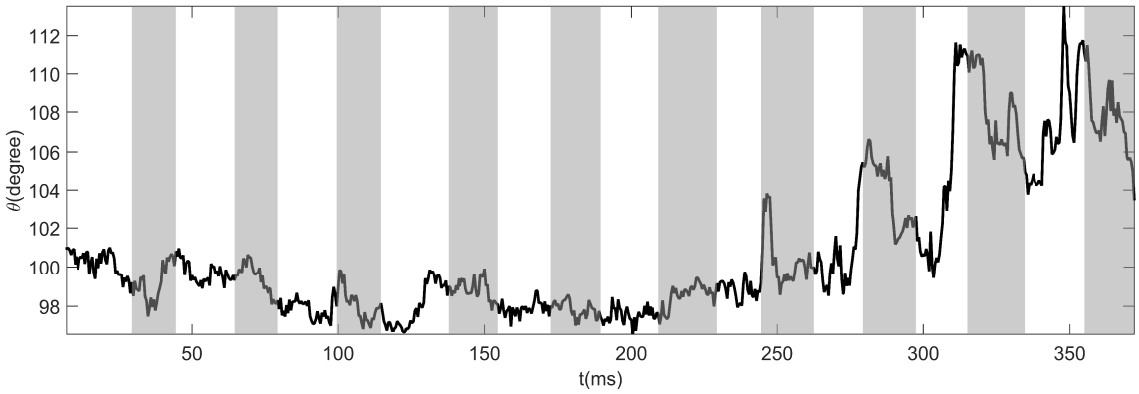
Figure 2.10: The finite element mesh generated by Gmsh.



(a) Time history of ρ

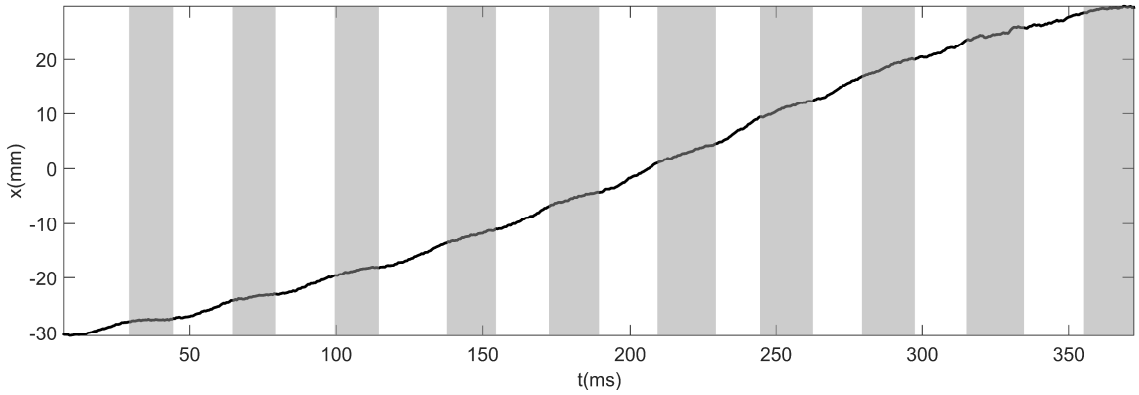


(b) Time history of ϕ

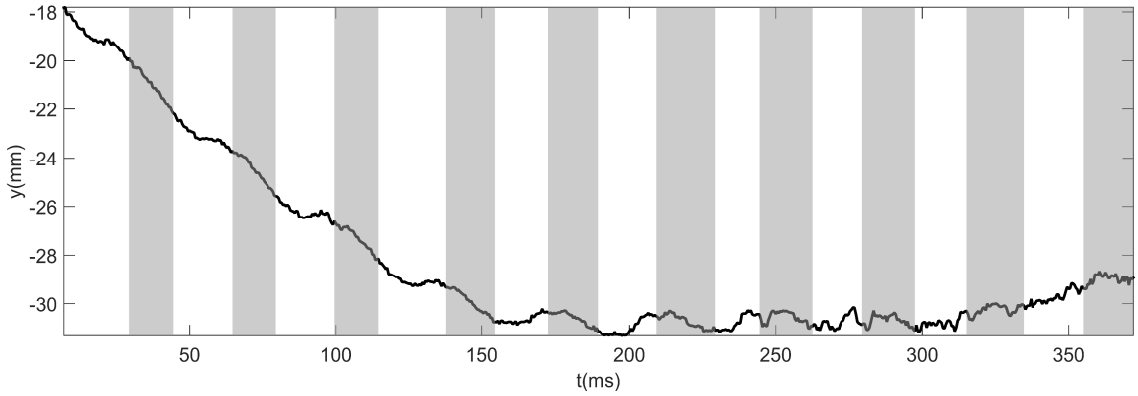


(c) Time history of θ

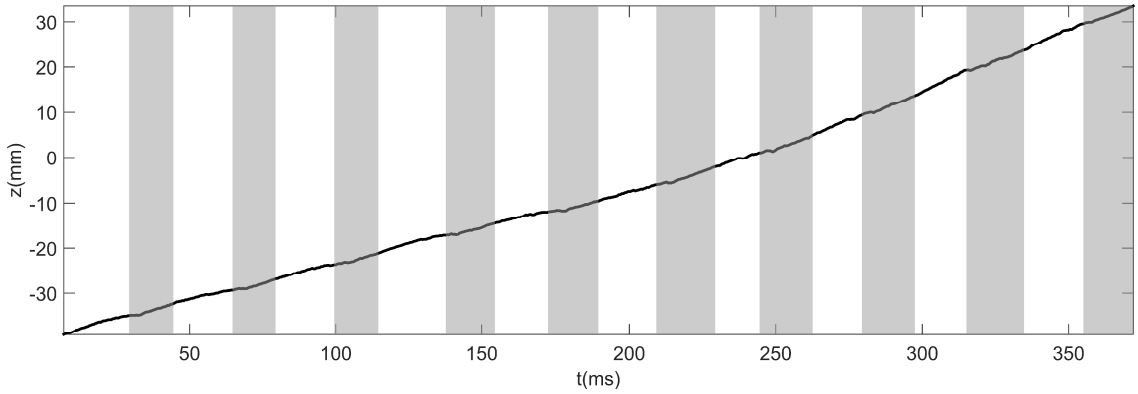
Figure 2.11: Time histories of Euler angles for moth body.



(a) x versus t

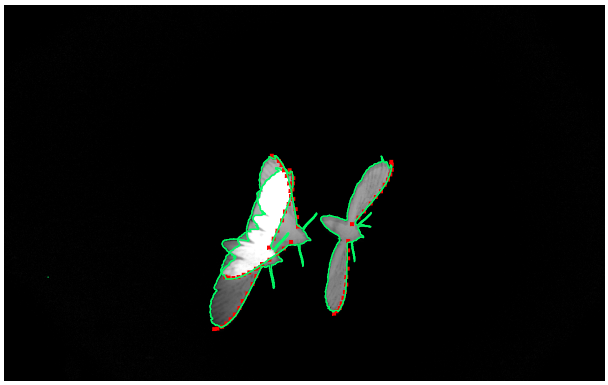


(b) y versus t

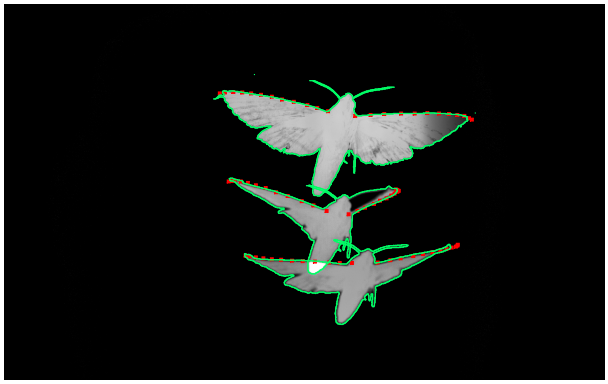


(c) z versus t

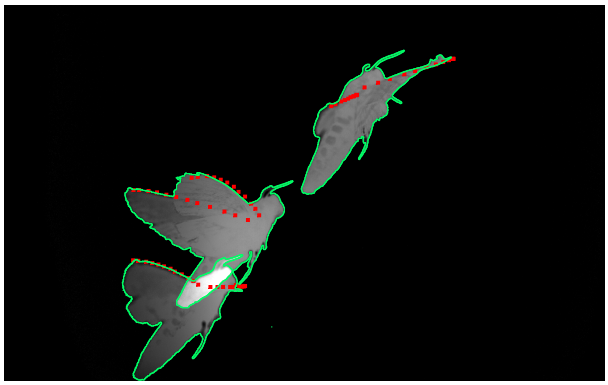
Figure 2.12: Time histories for position of geometric center of the moth body.



(a) View from camera XY

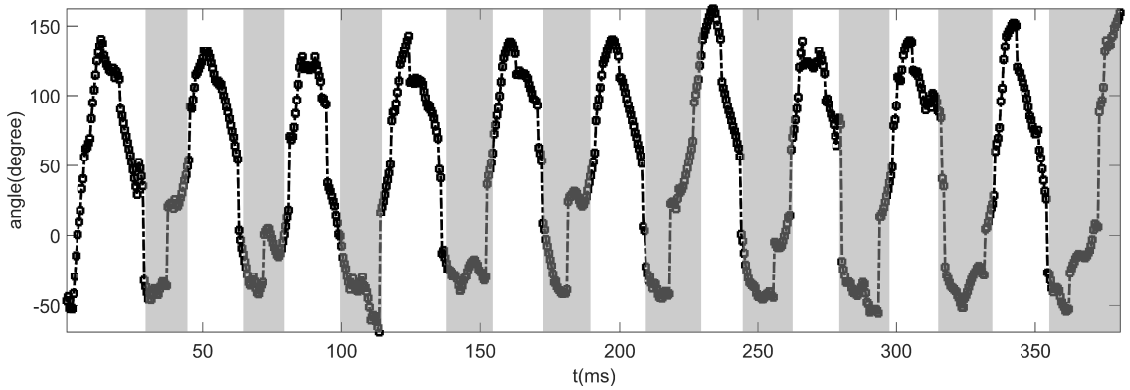


(b) View from camera XZ

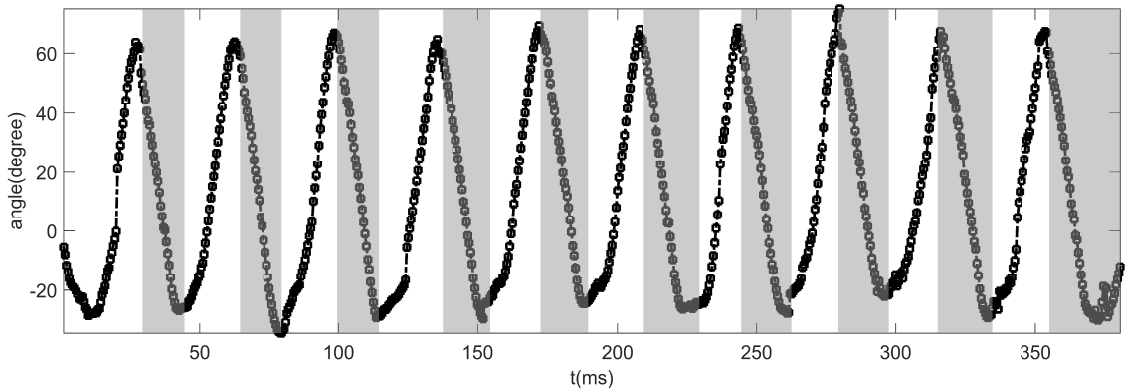


(c) View from camera YZ

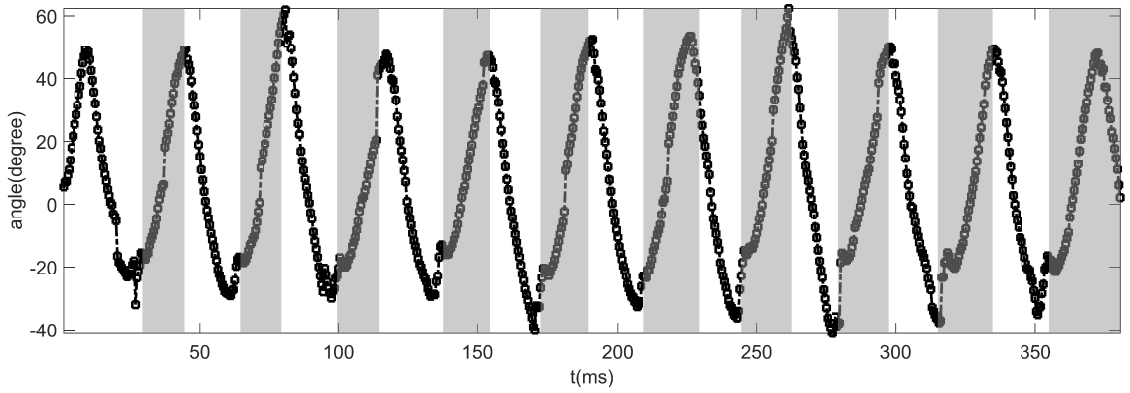
Figure 2.13: Representative frames of leading edge motion tracking. The red points show the estimation of the projections of the leading edge locations along with the raw images and boundary points.



(a) Time history of ρ

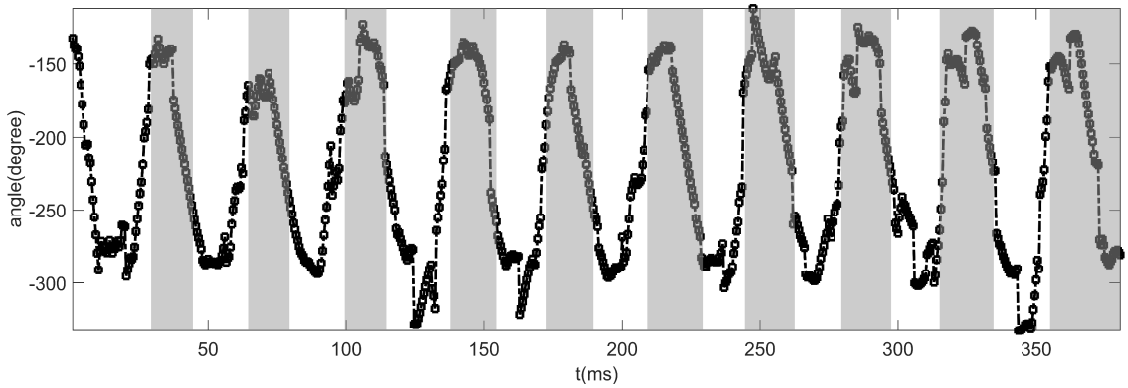


(b) Time history of ϕ

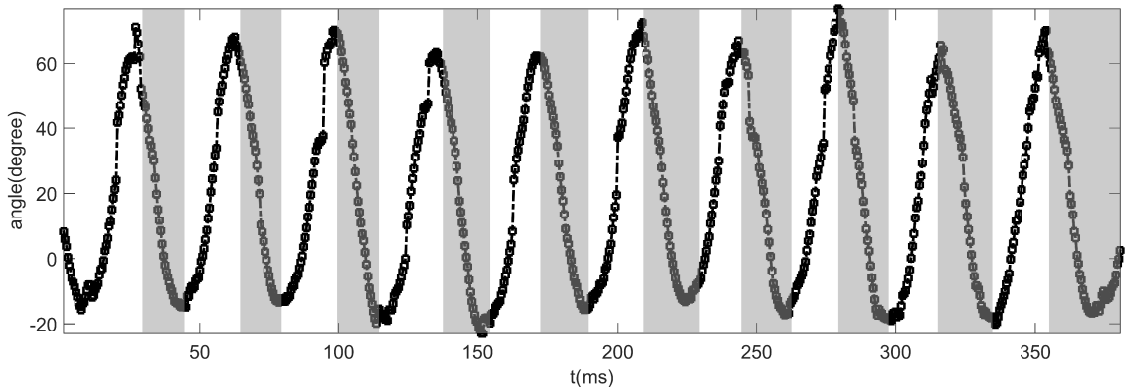


(c) Time history of θ

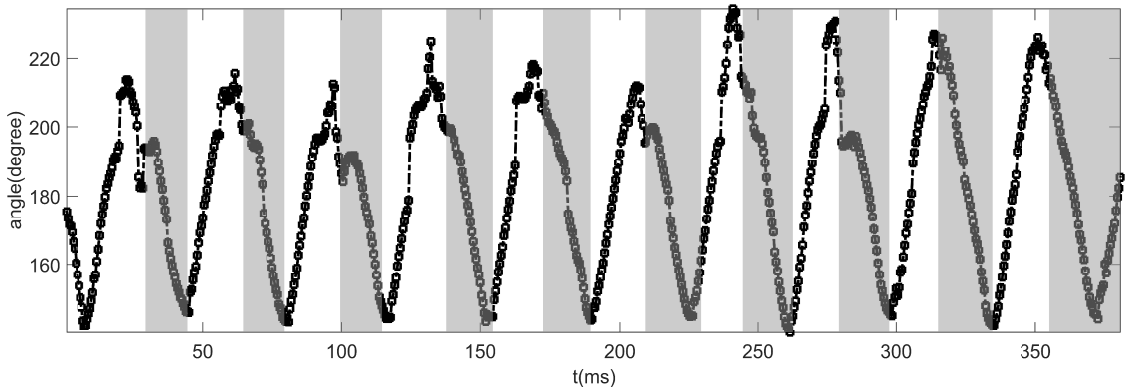
Figure 2.14: Time histories of Euler angles of leading edge of right wing.



(a) Time history of ρ

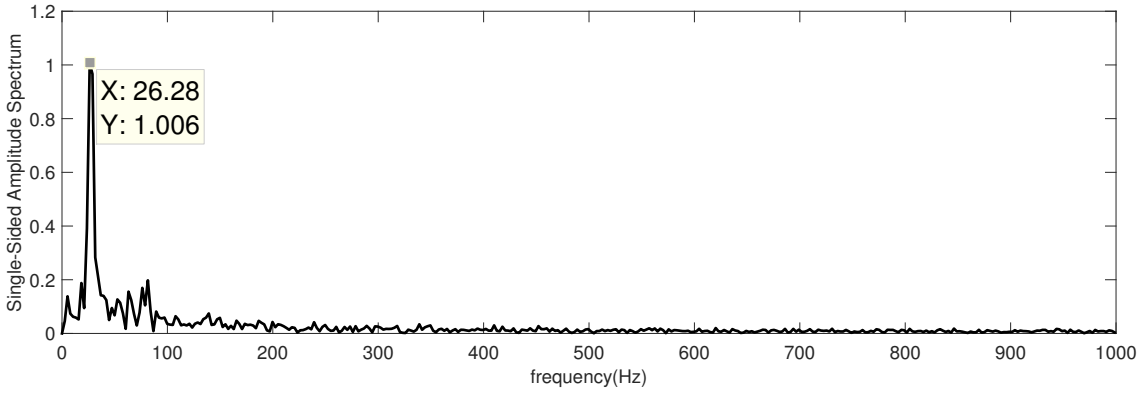


(b) Time history of ϕ

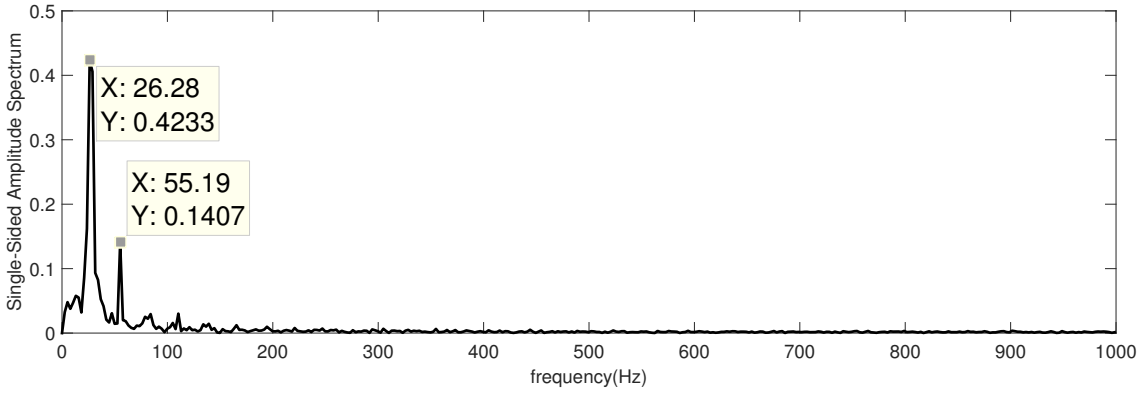


(c) Time history of θ

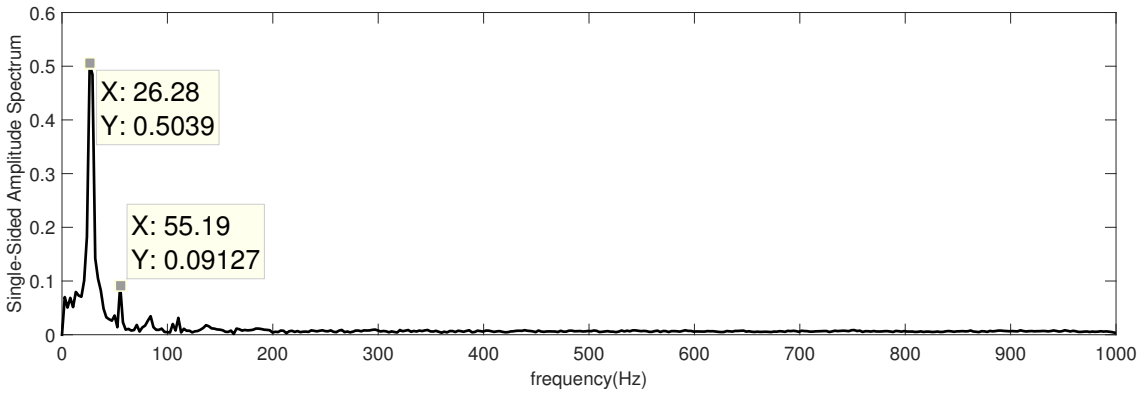
Figure 2.15: Time histories of Euler angles of leading edge of left wing.



(a) ρ

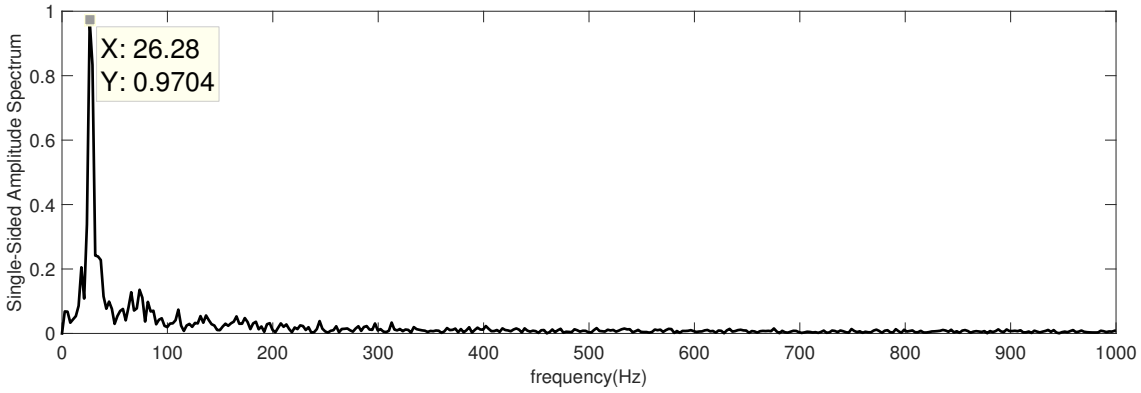


(b) ϕ

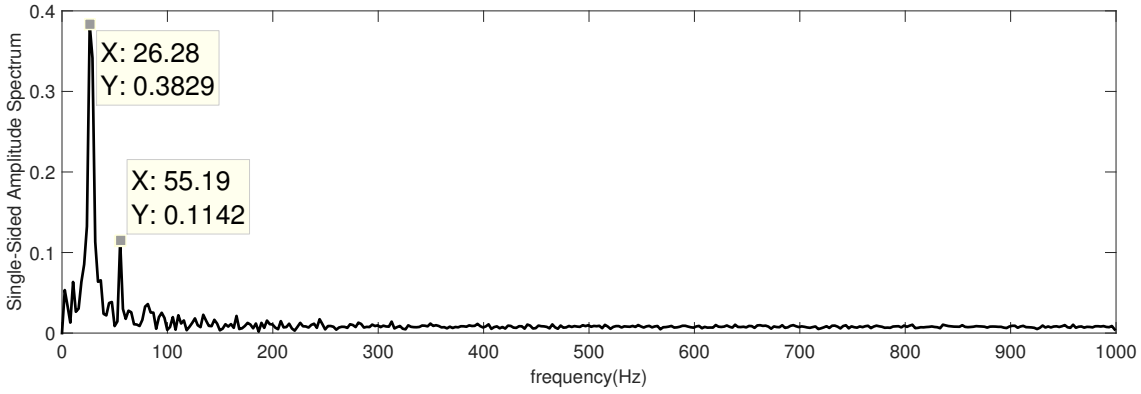


(c) θ

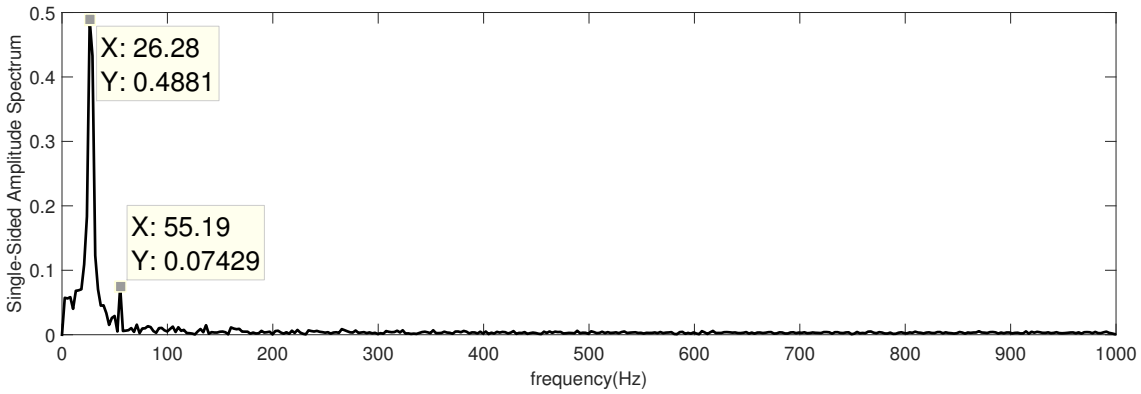
Figure 2.16: Fourier spectra of Euler angle histories for right wing leading edge.



(a) ρ



(b) ϕ



(c) θ

Figure 2.17: Fourier spectra of Euler angle histories for left wing leading edge.

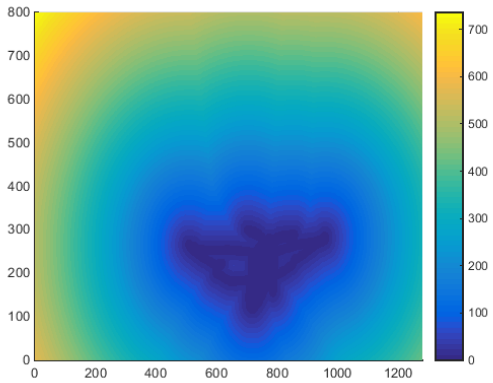
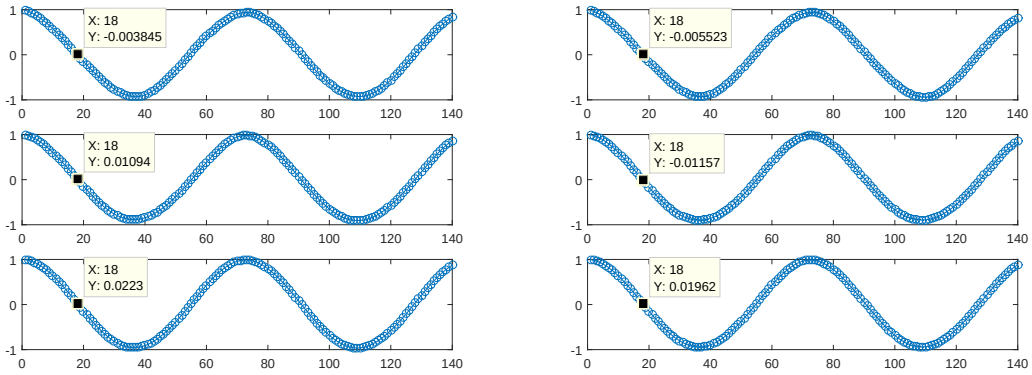
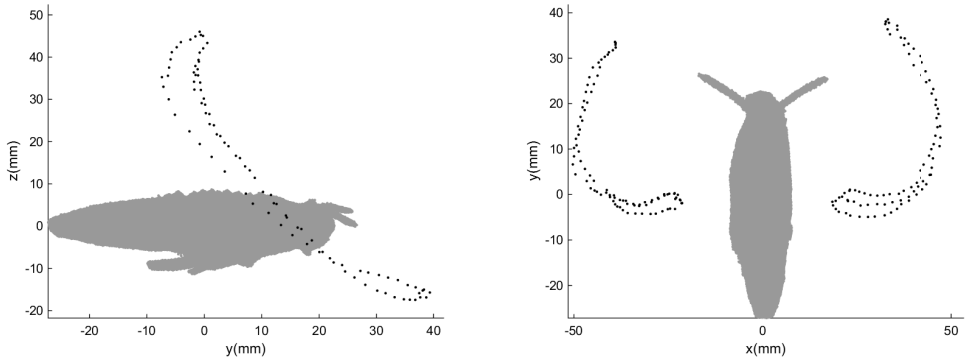


Figure 2.18: Distance map of each image point to the moth contour points and the body part. The domain of the map includes the entire image.



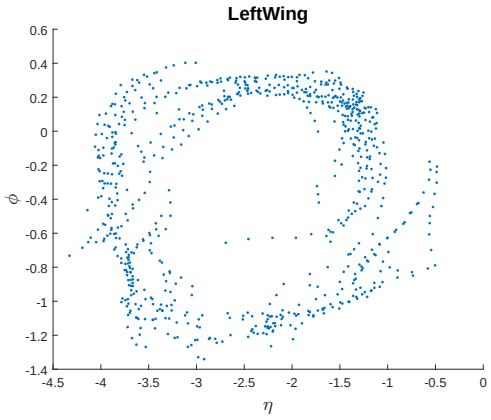
(a) Left wing (b) Right wing

Figure 2.19: Autocorrelation function $k - C(k\tau_s)$ plot for each Euler angle of each wing.

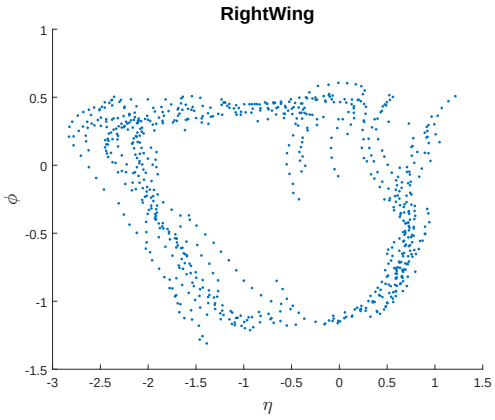


(a) Side view of the right wing (b) top view

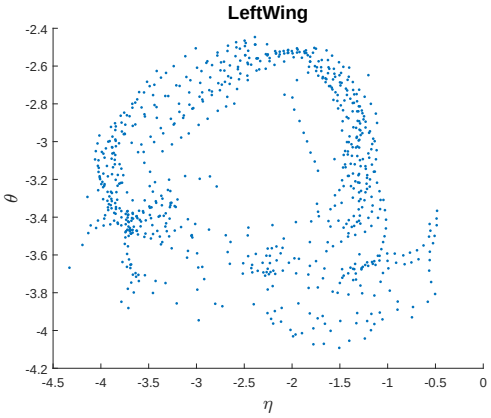
Figure 2.20: Representative cycle of the trajectory of the wing tip motion in B-frame.



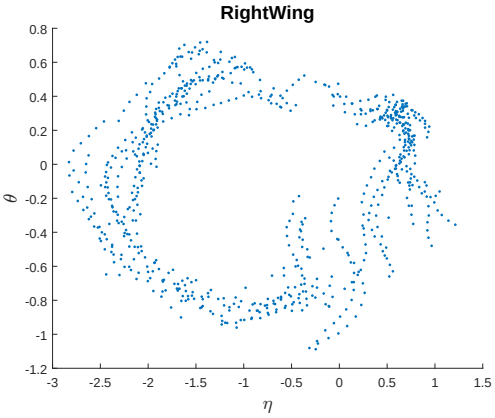
(a) $\rho - \phi$ plot



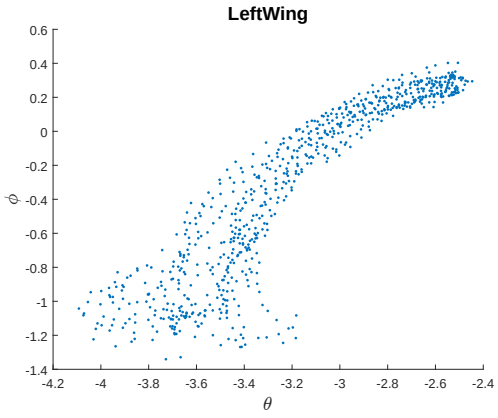
(b) $\rho - \phi$ plot



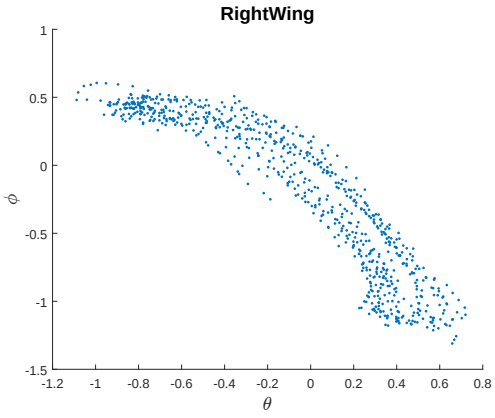
(c) $\rho - \theta$ plot



(d) $\rho - \theta$ plot

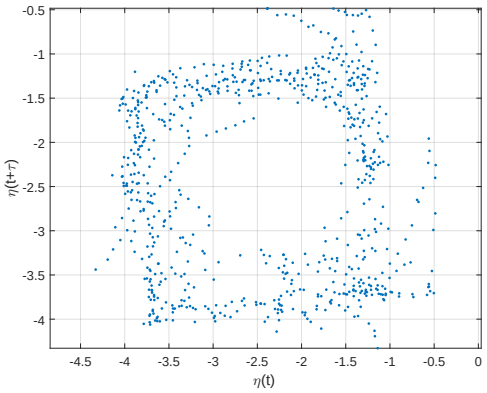


(e) $\theta - \phi$ plot

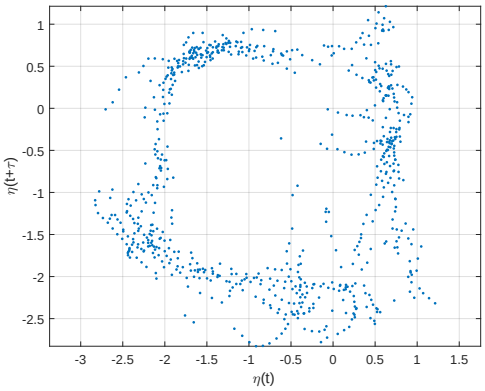


(f) $\theta - \phi$ plot

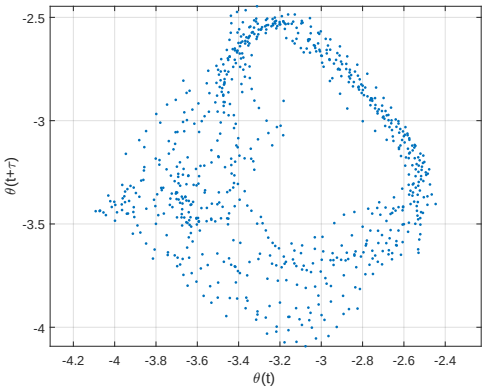
Figure 2.21: Phase portraits based on Euler angles for leading edge tracking.



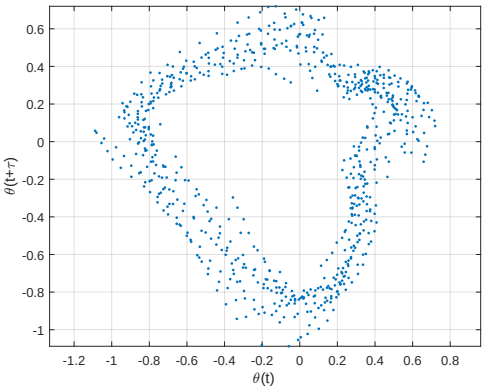
(a) Left wing. $\rho(t) - \rho(t+\tau)$ plot



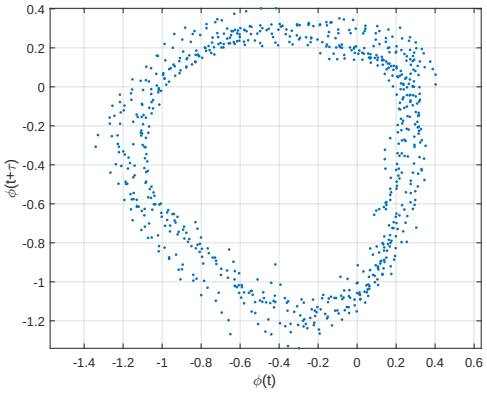
(b) Right wing. $\rho(t) - \rho(t+\tau)$ plot



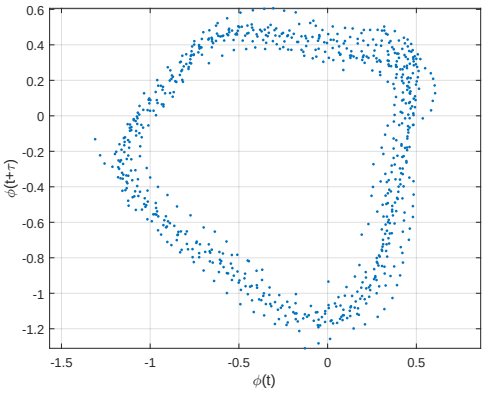
(c) Left wing. $\theta(t) - \theta(t+\tau)$ plot



(d) Right wing. $\theta(t) - \theta(t+\tau)$ plot



(e) Left wing. $\phi(t) - \phi(t+\tau)$ plot



(f) Right wing. $\phi(t) - \phi(t+\tau)$ plot

Figure 2.22: Pseudo-phase portraits for Euler angles from leading edge tracking.

Chapter 3: Motion reconstruction and estimation

Non-invasive motion reconstruction from multi-image data has received considerable attention. For a target object whose point-wise location and displacement field are defined continuously, each vision sensor is used to measure the intensity values of the corresponding projection plane in discrete pixels. Without one-to-one correspondence between pixels across the cameras as the known-feature tracking approach, the estimation of the displacement field of the considered target object is an ill-posed problem. Other considerations include the following: i) occlusions of 3D object projected on planar vision sensor and ii) innate limitation of information available from vision. As an example, the following is mentioned. For a perfect spherical black ball, the tangential velocity of the rigid body motion cannot be observed regardless of the number of cameras used [28].

Configurations can be estimated by matching the image data with a predefined model. In order to find a unique solution, the problem is usually formulated as an optimization problem, with the cost function consisting of a similarity term and a smoothness term as a regulator. In the computer vision community, a commonly used regulator is of the Tikhonov type[29]. To the best of the author's understanding, in most efforts, both the first and second spatial derivatives of the displacement

field have been used. These derivatives serve as the regulator, and the first and second derivatives provide a measure of the internal energy of a thin material due to the tensional force and bending moment, respectively [30, 28, 31, 32, 33, 34]. Essentially, a configuration that fits the measurement data is sought, while ensuring that this configuration contains the least internal energy. Additional factors, such as geometrical interpretation, prior knowledge, or extra measurement sources, can also be included in this framework [35, 36, 37, 38, 39, 40]. Finite elements (FE) can be naturally embedded into such a framework. When compared with the finite difference method used in some of the abovementioned literature, the FE method can result in a continuous displacement field estimation. Furthermore, the internal energy and its spatial derivatives are intrinsically calculated in the FE method. In Cohen and Cohen's work[41], a blend of plate and membrane elements is used to estimate the configuration of surface material. It is also worth mentioning that FE mesh matching and deformation estimation scheme is widely used in medical imaging society to either construct a customized FE model or estimation the deformation of brain or other organs[42, 43]. However, it is noted that the deformation is relatively small compared to the deformation observed in wing flapping motion.

With regard to the problem of estimating large deformations and rotation associated with entire wing motions during insect flight, different formulations for different experiment scenarios are proposed by the author. The images are binarized in this work considering the image quality from high-speed cameras. Such a procedure provides fast and robust location information of the boundaries of the target object.

3.1 Problem formulation

Consider an object with the reference configuration

$$\mathbf{x},\ \mathbf{x} \in \Omega_0, \tag{3.1}$$

and with current configuration at time t:

$$\mathbf{r}(\mathbf{x}, t),\ \mathbf{x} \in \Omega_0 \tag{3.2}$$

or

$$\mathbf{r}(\mathbf{x}, t) = \mathbf{x} + \mathbf{u}(\mathbf{x}, t) \tag{3.3}$$

where $\mathbf{u}$ describes the displacement field of the object.

A snapshot of the object's motion is captured by a set of cameras. Through appropriate image processing, discrete points that sketch the outline of the target object can be obtained. For the sake of simplicity, the intensity field is first transformed into binary image, and the output becomes a point cloud, which is a set of data points. Let

$$\mathbf{Y}^m = \left\{ \mathbf{y}_1^m, \mathbf{y}_2^m, \ldots \mathbf{y}_i^m | \, \mathbf{y} \in R^2 \right\} \tag{3.4}$$

Here, $\mathbf{y}_i^m$ is the i^{th} image point captured from the m^{th} camera, describing the silhouette of the target object on the projection plane of the camera. The current

configuration is discretized as

$$\boldsymbol{Q}^m \mathbf{r}(\mathbf{x}, t) = \{\mathbf{y}_1^m + \boldsymbol{\epsilon}_1^m,\, \mathbf{y}_2^m + \boldsymbol{\epsilon}_2^m \ldots \mathbf{y}_i^m + \boldsymbol{\epsilon}_i^m\} \tag{3.5}$$

where Q is the coordination transformation operator from 3D space to the projection plane of the camera. At this stage, the inverse ill-posed problem that one has is the following: Given a set of measurements from the cameras of both reference and current configurations, find the true state of the current configuration. There does not exist a unique solution because there is no one-to-one correspondence between the measurement data and the current configuration. In other words, the quantity of $\boldsymbol{\epsilon}_i^m$ can not be obtained. Therefore, a regularization technique has to be introduced here in order to seek a unique solution. The most commonly applied regulator in such inverse problems in image processing is of the Tikhonov type, and this regulator has drawn analogies with the strain energy of the target object. Hence, instead of formulating a discrete Tikhonov regulator, strain energy estimated through the finite element method is directly used as the regulator.

The author considers the following formulation of problem as an illustrative example.

Minimize the strain energy s.t. $\mathbf{y}_i^m \in \boldsymbol{Q}^m \mathbf{r}^m(\mathbf{x}, t)$

In the static case, among all the possible solutions that can possibly be determined, the one configuration that is formed through the least power input to its reference configuration is chosen, as the strain energy is determined by the work

done by the external force.

3.1.1 Optimization tools

In this work, the author has proposed a multi-level optimization procedure : Initial optimization and local optimization. A reduced-order deformation model is adopted in the initial optimization step to efficiently locate the rough position of the global minimum. The pattern search function in the optimization toolbox provided in MATLAB is used to solve the initial optimization problem without evaluating the gradients of the cost function.

Local optimization is then solved by the gradient-based Broyden-Fletcher-Goldfarb-Shannon (BFGS) algorithm[44]. Details of each step are illustrated after in this thesis.

3.1.2 Initial optimization

Initial optimization is executed first for the sake of expediting the convergence rate. In this study, this step is actually necessary in order to locate the global minimum with available computational power. Consider a point on the wing

$$^{R'}\mathbf{r} = \mathbf{R}(\rho, \phi, \theta)^{W}\mathbf{r} + \mathbf{d}^{W/O} \tag{3.6}$$

The position vector can be described by the rigid-body motion of the W-frame and the deformation on the W-frame. Here, the author uses the linear mode shapes as a basis to describe the deformation of the wing. The linear mode shapes can be

obtained by solving the eigenvalue problem of

$$\left\{\mathbf{K} - \omega_n^2 \mathbf{M}\right\} \boldsymbol{\phi}_n = \mathbf{0} \tag{3.7}$$

where $\mathbf{K}$ and $\mathbf{M}$ are the constant stiffness and mass matrix, respectively. ω_n and $\boldsymbol{\phi}_n$ represent the natural frequency and the mode shape, respectively. The deformation is then assumed to be

$$\begin{aligned} {}^{W}\mathbf{r} &= \begin{bmatrix} \boldsymbol{\phi}_1({}^{W}\mathbf{r}) & \boldsymbol{\phi}_2({}^{W}\mathbf{r}) & \dots & \boldsymbol{\phi}_n({}^{W}\mathbf{r}) \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_n \end{bmatrix} \\ &= \boldsymbol{\Phi}({}^{W}\mathbf{r})\mathbf{p} \end{aligned} \tag{3.8}$$

where $\mathbf{p}$ are the modal participation factors.

The rigid body motion is then acting on the locally deformed body. Therefore, there are $n + 6$ degrees of freedom , which serve as optimization variables, that is

$$\min_{(\mathbf{p},\rho,\phi,\theta,\mathbf{d}^{W/O})} D \tag{3.9}$$

where D is the projection difference as aforementioned. ρ, ϕ, θ and $\mathbf{d}^{W/O}$ are the Euler angles and the displacement of the W-frame in R'-frame, respectively.

Note that the mode shapes may not necessarily be orthogonal to each other, although if so, one could take advantage of this property to find a unique solution to

the optimization problem. The chosen spatial shapes can be collected for different settings such as stiffness distribution if the real structure is more complicated than the reference model. Realistic loads could be applied to obtain various shapes as well. The ultimate purpose is to qualitatively describe the principal deformation observed in the data.

The author would like to empathize that, for large deformation which is the considered case of interest, the magnitudes of the linear mode shapes estimated by matching the measured images are out of the physical range of linear approximation. Therefore, the refined optimization described in the next section must be executed after this step. It is observed that the positions of internal nodes would be adjusted, and the strain energy is relaxed after the refined optimization process.

3.1.3 Refined optimization

The chosen objective function has the form of soft constraint problem

$$\min_{\mathbf{q}} \pi = \min_{\mathbf{q}} (\alpha D + U) \tag{3.10}$$

instead of the formulation of a hard constraint problem for the sake of tackling the problem practically. Here, D is called the similarity penalty, whose gradient serves as the image force. α is a weighting factor, and the internal energy U represents the smoothness penalty.

The dual objectives are to fit the FE body over the point cloud and to minimize the energy involved to make the deformation realistic. Although a FE model has

finite degrees of freedom as a finite difference method does, a FE model can be used to obtain a continuous displacement field estimation. Furthermore, the strain energy and its gradient and Hessian of the FE body can be internally computed.

3.2 Computations

3.2.1 Similarity penalty

In this section, various ways to formulate the similarity penalty from binary images are explored. The properties of each formulation are discussed, and the feasibility for the use of wing motion estimation will be shown in later section. Although the feature extraction technique has not been introduced yet, it is noted that it can be used in an auxiliary role to further enhance the the motion estimation technique depicted in this work.

3.2.1.1 Silhouette cone intersection

Visual hull reconstruction can be applied to multiple two-dimensional images to reconstruct a three-dimensional voxel cloud by neglecting the perspective distortion [45]. The concept is based on taking the intersection of the extrusion of each image on its third dimension. To fit the voxel cloud to the reference body, the author defines the similarity term as the sum of the square distance of each voxel point to its closest point on the reference body; that is,

$$D = \sum_{i=1}^{N} \|\mathbf{r}_i(\mathbf{y}_i) - \mathbf{y}_i\|^2 \tag{3.11}$$

Here, $\mathbf{y}_i \in R^3$ represents the voxel cloud and $\mathbf{r}_i(\mathbf{y}_i)$ is the registered point on the reference body of the voxel point. However, the silhouette cone intersection method produces the maximum volume representation of the images. The result of the reconstruction can be erroneous while the target object experiences large rotations and large deformations. An illustrative example is shown in Figure 3.1. Although it has been shown that the rigid-body kinematics is captured well from the voxel cloud [25], the volumetric error is no longer negligible for the large motions considered here.

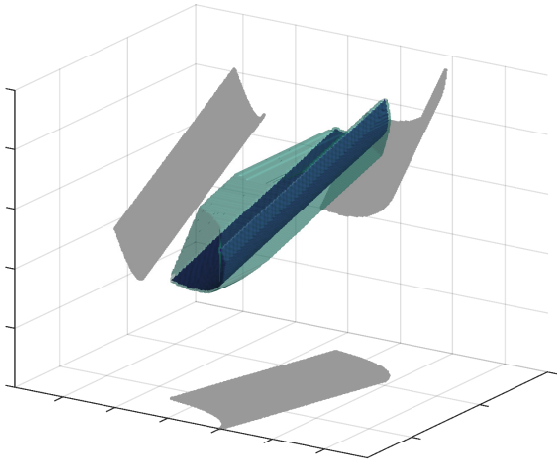


Figure 3.1: A rotated U-shape thin sheet (blue) and the maximum volume representation (green) constructed from the sheet's shadows in three orthogonal planes.

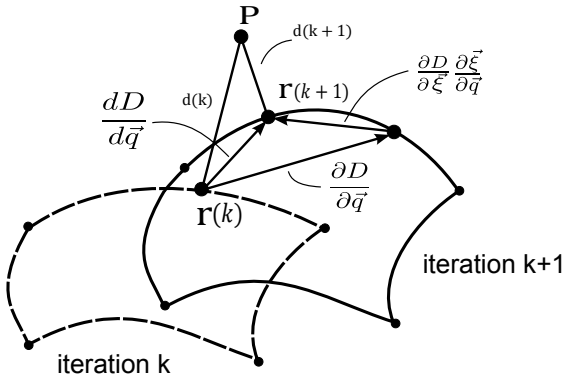


Figure 3.2: Schematic plot of total derivative of the closest distance between an image point P to the closest point on an element with respect to nodal displacement.

3.2.1.2 Area similarity

Consider a set of silhouettes captured by multiple cameras. Evaluating the consistency between the projection of FE body and the shadow captured from the cameras provides an intuitive idea of exploiting information from the silhouettes. The similarity term can be simply computed by counting the pixel points that are

not overlapping. The author also suggests to normalize the similarity term of each camera by the number of active pixels, so the cost function penalizes the similarity term on each camera equally, and after a few arithmetic operations, the similarity terms can be interpreted as the mean percentage projection differences.

However, the gradient of such a function is hard to obtain analytically. A brute-force search or numerical gradient evaluation scheme may need to be used; either way, the solution procedure can be computationally challenging. For using finite difference calculation, the step sizes are carefully chosen because of the discrete property of images.

3.2.1.3 Distance penalty

Following the spirit of direct penalty on two-dimensional images, the closest distance of image point to the corresponding reference body is penalized instead in order to reduce the computational expense. Such formulation relies on the bundle-adjustment philosophy.

The cost function takes the form

$$D = \sum_m D_m = \sum_m \sum_{i=1}^{N_m} \|\boldsymbol{Q}^m \mathbf{r}_i^m(\mathbf{y}_i^m) - \mathbf{y}_i^m\|^2 \tag{3.12}$$

This cost function shares the same minima as the function used for the area penalty. The shape difference between the image and the FE model is quantified as the sum of closest Euclidean distance of each image point to the projected FE surface. Compared with area difference, the weighting of each pixel is scaled intuitively by

the registration between each image point to the reference model, and the gradient can be evaluated as discussed below. The camera setup is assumed well-calibrated, and the images are the shots from three orthogonal principal directions. Perspective distortion is also neglected for the sake of simplicity.

Consider an image point $\mathbf{P}$ and the registered position on the corresponding element. Given the FE information about surface element e, and the tuple representing the position of the pixel $\mathbf{p}$, the minimum distance can be stated as the solution of

$$ {}^{e}d^2 = \min_{(\xi,\,\eta)} \mathbf{d}\cdot\mathbf{d} = \min_{(\xi,\,\eta)} (\mathbf{p}-\mathbf{r})\cdot(\mathbf{p}-\mathbf{r}) \tag{3.13a} $$

$$ \xi \in [-1,\,1] \tag{3.13b} $$

$$ \eta \in [-1,\,1]. \tag{3.13c} $$

The bounds on the natural coordinates keep the point on the surface element. Here, the position $\mathbf{r}$ in an element is approximated by using the standard finite element description of the surface

$$ \mathbf{r} = \mathbf{N}(\xi,\,\eta)\cdot(\mathbf{x}+\mathbf{q})\,. \tag{3.14} $$

where $\mathbf{x}$ and $\mathbf{q}$ is the tuple of reference configuration and displacements associated with the degree of freedoms, respectively. The matrix $\mathbf{N}$ consists of the shape functions. The optimization variables used to fit the FE over the pixel cloud are the displacements $\mathbf{q}$ of nodes of the mesh. Therefore the gradient of the similarity

function from one camera that needs to be computed is

$$\frac{\partial D_m}{\partial \mathbf{q}} = \sum_{i=1}^{N_m} \frac{\mathrm{d}}{\mathrm{d}\mathbf{q}} d_i^2 \tag{3.15}$$

which can be computed by pixel, and then assembled into the global tuple. Looking at this at the component level, one can employ the total derivative

$$\frac{\mathrm{d}}{\mathrm{d}q_k} d_i^2 = \frac{\partial d_i^2}{\partial q_k} + \frac{\partial d_i^2}{\partial \xi}\frac{\partial \xi}{\partial q_k} + \frac{\partial d_i^2}{\partial \eta}\frac{\partial \eta}{\partial q_k} = \frac{\partial d_i^2}{\partial q_k} + \frac{\partial d_i^2}{\partial \boldsymbol{\xi}} \cdot \frac{\partial \boldsymbol{\xi}}{\partial q_k} \tag{3.16}$$

The extra terms are due to the sensitivity of the of the closet point $\bar{\boldsymbol{\xi}} = (\bar{\xi}, \bar{\eta})$ to the perturbation of ∂q_k, as illustrated in Figure 3.2. Each of the terms in equation (3.16) will be constructed as discussed in Subsection A.4.1.

3.2.2 Strain energy

The strain energy in the body can be defined as

$$U_s = \frac{1}{2}\int_{\Omega} \boldsymbol{\sigma} : \boldsymbol{\varepsilon} \, \mathrm{d}V. \tag{3.17}$$

This can be transformed back to the reference configuration to make the implementation simpler. Recall that the true stress and Euler strain are related to the second

Piola-Kirchhoff stress and Green strain by the well-known identities [46]

$$\mathrm{d}V = \det\boldsymbol{F}\ \mathrm{d}V_0 \tag{3.18a}$$

$$\boldsymbol{\varepsilon} = \boldsymbol{F}^{-T}\boldsymbol{E}\boldsymbol{F}^{-1} \tag{3.18b}$$

$$\boldsymbol{\sigma} = (\det\boldsymbol{F})^{-1}\boldsymbol{F}\boldsymbol{S}\boldsymbol{F}^{T}. \tag{3.18c}$$

Therefore,

$$\begin{aligned}\int_{\Omega} \boldsymbol{\sigma} : \boldsymbol{\varepsilon}\ \mathrm{d}V &= \int_{\Omega} \mathrm{tr}\left(\boldsymbol{\sigma}^T\boldsymbol{\varepsilon}\right)\ \mathrm{d}V \\ &= \int_{\Omega_0} (\det\boldsymbol{F})^{-1}\mathrm{tr}\left(\boldsymbol{F}\boldsymbol{S}^T\boldsymbol{F}^T\boldsymbol{F}^{-T}\boldsymbol{E}\boldsymbol{F}^{-1}\right)\det\boldsymbol{F}\ \mathrm{d}V_0 \\ &= \int_{\Omega_0} \mathrm{tr}\left(\boldsymbol{S}^T\boldsymbol{E}\right)\ \mathrm{d}V_0.\end{aligned}$$

Here, in the last step, the invariance of the trace to similarity transform is used. This implies that the total strain energy can also be computed through the use of FE as

$$U_s = \frac{1}{2}\int_{\Omega_0} \boldsymbol{S} : \boldsymbol{E}\ \mathrm{d}V_0 = \frac{1}{2}\sum_{e=1}^{n_{el}} \int_{\Omega_0^e} \boldsymbol{S} : \boldsymbol{E}\ \mathrm{d}V_0. \tag{3.19}$$

This permits a direct calculation in the total Lagrangian based FE model, based on materials such as Biot or Kirchhoff materials.

3.2.3 Constraints

3.2.3.1 Incompressibility

For the particular future application of simulating the fluid-body interactions using the estimated motion here, the total volume of the body must be fixed. This is not the usual "incompressibility" condition since that it is point-wise. Computation of the initial volume of the mesh is done by summing over the volume of each element

$$V_0 := \sum_{e=1}^{n_{el}} \int_{\Omega_0^e} \mathrm{d}V_0 \tag{3.20}$$

Once the body has been deformed, the current configuration needs to be used to compute the volume

$$V := \sum_{e=1}^{n_{el}} V^e = \sum_{e=1}^{n_{el}} \int_{\Omega^e} \mathrm{d}V \tag{3.21}$$

Examining the volume of a single deformed element, it can be cast as an integral in the reference configuration as

$$V_e = \int_{\Omega^e} \mathrm{d}V = \int_{\Omega_0^e} \det\boldsymbol{F}\ \mathrm{d}V_0. \tag{3.22}$$

Here, $\boldsymbol{F}$ is the deformation gradient tensor, for which one employs the well-known identity of the Jacobian determinant [46, Eq 2.50]. The residual constraint ($h = 0$)

can be stated simply as

$$h(\mathbf{q}) = V(\mathbf{q}) - V_0 \tag{3.23}$$

3.2.3.2 Non-folding condition

The size of the iterative descent step is limited in order to avoid non-physical configuration, such as folding and flipping. For the Jacobian of the isoparametric mapping of the e^{th} element and it's determinant is such that:

$$\det(\mathbf{J}^e) = \det\left(\frac{\partial \mathbf{r}}{\partial \boldsymbol{\xi}}\right) > 0 \tag{3.24}$$

Inspections over all the Jacobians of the transformation of the finite element body is examined while a pseudo step is attempted. The step is uniformly scaled down if the attempted step causes the folding condition.

It is noted that a similar technique is applied to prevent the FE body from folding in the area of mesh registration. Unlike the study of Bucki et al. [43], wherein the use of the deformation gradient tensor is suggested, here, computation of the Jacobian is more efficient in the finite element framework.

3.3 Case study

The algorithm is evaluated by using the results presented in Fitzgerald's work [22] as a test case. In Fitzgerald's work, fluid-structure interactions associated with

a non-linear cantilever plate under large flapping motion prescribed at the root positions has been studied. The same material and element type are used here in the wing FE model. The deformation of the plate is studied here as an illustrative case of what can be can done with experimental data for flapping wings. The flow chart for this case study is shown in Figure 3.3. One snapshot of time of the deformation information is projected onto three orthogonal planes to mimic image data. The dimension of the plate is $l \times 0.3l \times 0.05l$ in nondimensional lengths, with the maximum deformation approximately equalling $0.28l$.

The initial optimization step is omitted in this test because the configuration of the plate is rather simple. Some snapshots of iteration steps in the refined optimization procedure carried out by using the distance penalty are shown in Figure 3.4. The true configuration, the deformed body, their corresponding projections, and the negative gradients are plotted.

Note that the formulation of the cost function in which area penalty serves as the similarity term is also tested in the same manner. A similar convergence rate in iteration steps is observed, while it took more than 20 times longer computational time than the distance penalty on the same machine.

3.4 Application to free flight data of hawkmoth

The motion estimation scheme is next applied to the free flight data of hawkmoth, with the customized reference FE wings and the images where the pixels belonged to the body are activated already. In other words, the similarity penalty

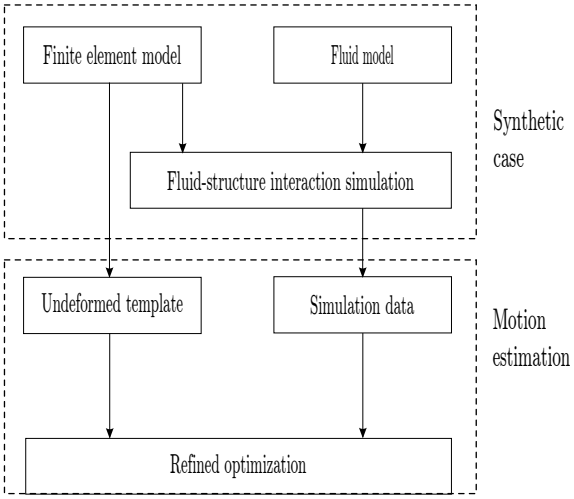


Figure 3.3: Flow chart for motion reconstruction of nonlinear cantilever beam undergoing large motion.

is used to penalize the projection difference between the captured image and the body images, which the deformable reference wing FE projected onto.

Initial optimization is executed first following the procedure described in Subsection 3.1.2. The initial condition for the optimization procedure is zero deformation, with the rigid-body motion that is estimated in Subsection 2.5.3. From the experiments with hawkmoth wings by Sims [1] and Fitzgerald [22], the first three linear mode shapes are acquired. As reported by Sims [1], the first three mode shapes correspond to bending motion in span-wise direction, torsional motion, and bending motion in chord-wise direction, respectively, and the natural frequencies in air are as listed in Table 3.1. As a starting point, the FE reference model of the wing is just one piece of planform with uniform thickness using a custom 27-node hexahedral cubic element[22]. One can extend this work by constructing a more sophisticated FE model by defining the veins and the mechanism between the forewing and hindwing, for example. The linear mode shapes are acquired by solving the eigenvalue

	ω(Hz)	ω_f/ω
mode 1	58.75	0.47
mode 2	75	0.36
mode 3	95	0.29

Table 3.1: Damped natural frequencies reported by Sims[1] and the nondimensional ratios with the flapping frequency estimated from the leading edge motion.

problem with various stiffness distributions. The clamped-free essential boundary condition is applied to not only mimic the experiments by Sims[1], but also used as a realistic choice for fitting the root-actuating flapping motion.

Local optimization step is executed on each time frame independently afterwards on the High Performance Computing Cluster at the University of Maryland. The result of mean projection difference for each time step is shown in Figure 3.6a, with the mean projection difference around 7%. The improvement of similarity from initial optimization to local optimization step is also shown in Figure 3.6b; selected frames of estimated motion in 3D and the projections are shown in Figure 3.7 and Figure 3.8.

A selective frame of the residue of the strain energy after the optimization procedure is shown in Figure 3.9. Ideally speaking, the strain distribution is "optimal" for the specific deformation. However, high strain is observed on the edge of the wing due to high image force caused by the imperfectness of the reference wing geometry. Note that the high strain at the intersection of the hindwing and the forewing may not be realistic, since they are actually hooked together.

To summarize, the flow chart of the procedure from recording the free flight

data in Chapter 2 to motion reconstruction is shown in Figure 3.5.

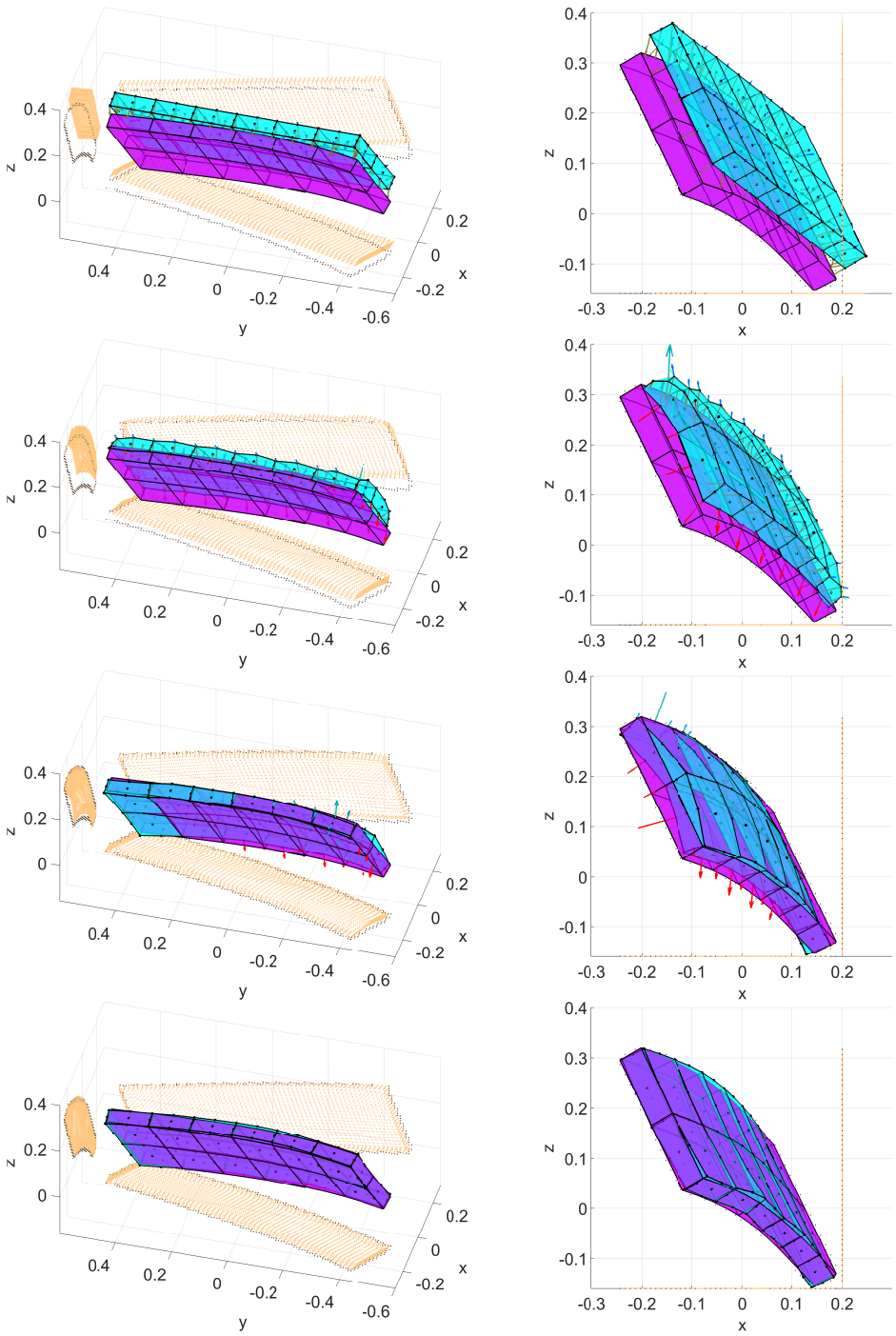
3.4.1 Kinematic analysis

The rigid-body motion of the root node is shown in Figure 3.12; this represents the flapping input to the passive wing structure. The results of application of the FFT to the time history of each Euler angle are shown in Figure 3.13. Compared to the results obtained from the leading edge tracking in Figure 2.16, the fundamental frequency is found to be higher; this is because the leading edge tracking technique is meant to capture rigid-body motion, whereas the actual motion actually consists of the rigid-body motion and significant torsional deformation. Thus, the fundamental frequency estimated from leading edge tracking should be considered as the effective operational frequency of the leading edge instead of the input frequency.

The author would also like to point out that the higher harmonic frequency at the elevation angle of the root motion appears to be more significant than it is in the case of leading edge motion tracking. The higher harmonic characteristic may be over-compensated by the deformation, since the higher harmonic frequency is more than two times the fundamental frequency.

With the estimation of the entire wing motion, here, the author only plots the trajectories of the tip, the root, and the trailing edge to gain insights into the flight mechanism. A representative cycle of wing beat in B'-frame is plotted in Figure 3.10. Amplitudes of the oscillation are calculated by the absolute distance between the current position to the mean coordinate of the trajectory. The am-

plitudes of oscillation of the left wing and right wing are shown in Figure 3.11. It can be clearly seen that there exists a phase lag between the tip motion and the trailing edge motion. Although the root motion is rather noisy because of the sloppy reference model currently used, phase lead is also observed in the root motion. This phase difference is expected, and it is attributed to the damping effects in the flow and the material.



(a) A 3D view

(b) View from camera XZ

Figure 3.4: Snapshots from selected angles are shown of iteration steps 0,1,8, and 123 involved in the optimization procedure. The purple body is the true configuration with the black projected outline being made up of the synthetic image points. The blue body is the reference body, which is taken to the deformed state through the optimization procedure with brown solid silhouettes. The blue, red, and green arrows are the negative directions of the strain penalty, the similarity penalty, and the gradient, respectively.

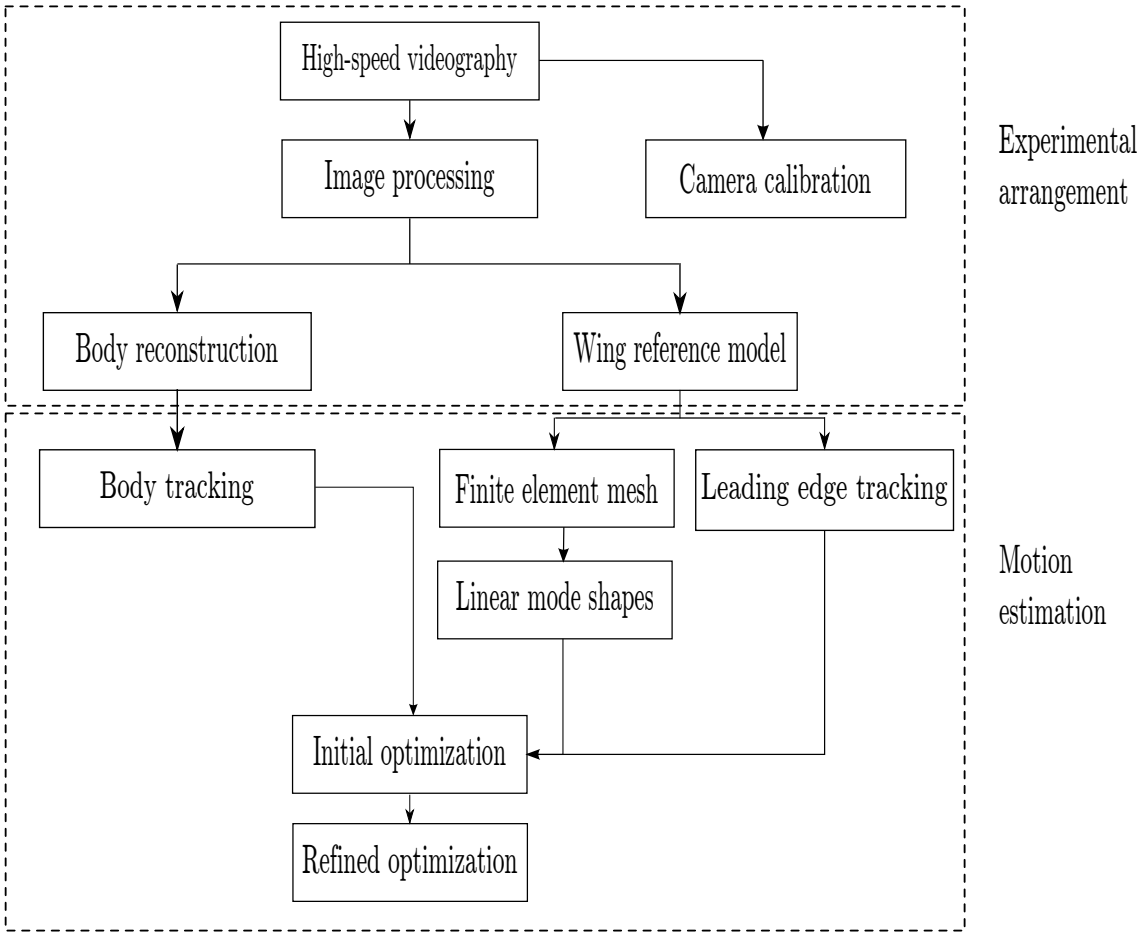
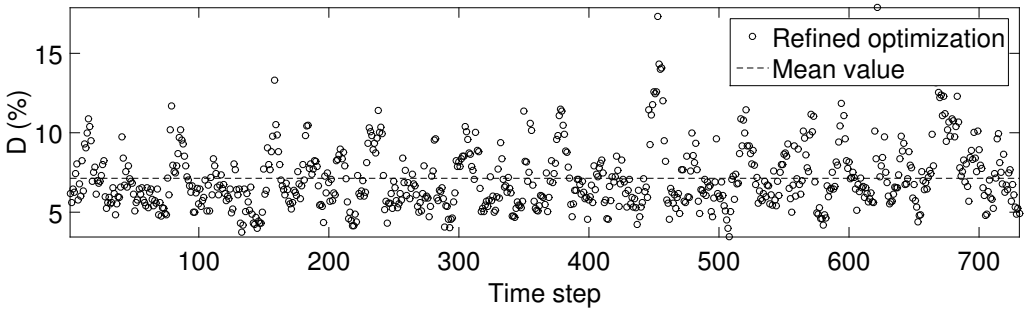
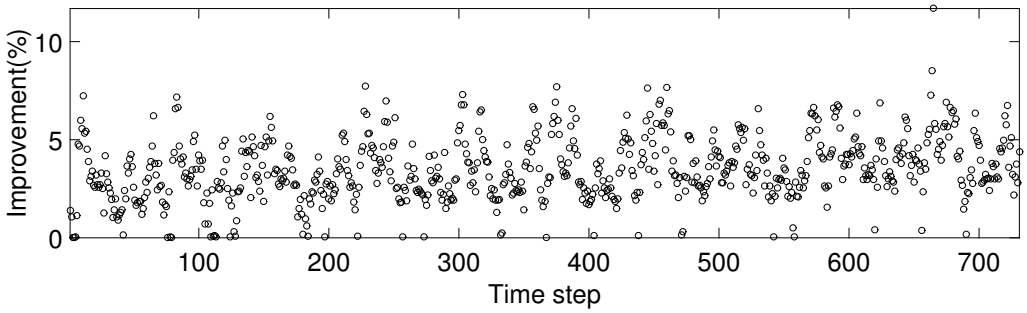


Figure 3.5: Flow chart of motion reconstruction of hawkmoth from high-speed videography.



(a) Mean projection difference at each time step



(b) Improvement of mean projection difference at each time step from global optimization to local optimization

Figure 3.6: Evaluation of the projection difference (%) at each time step.

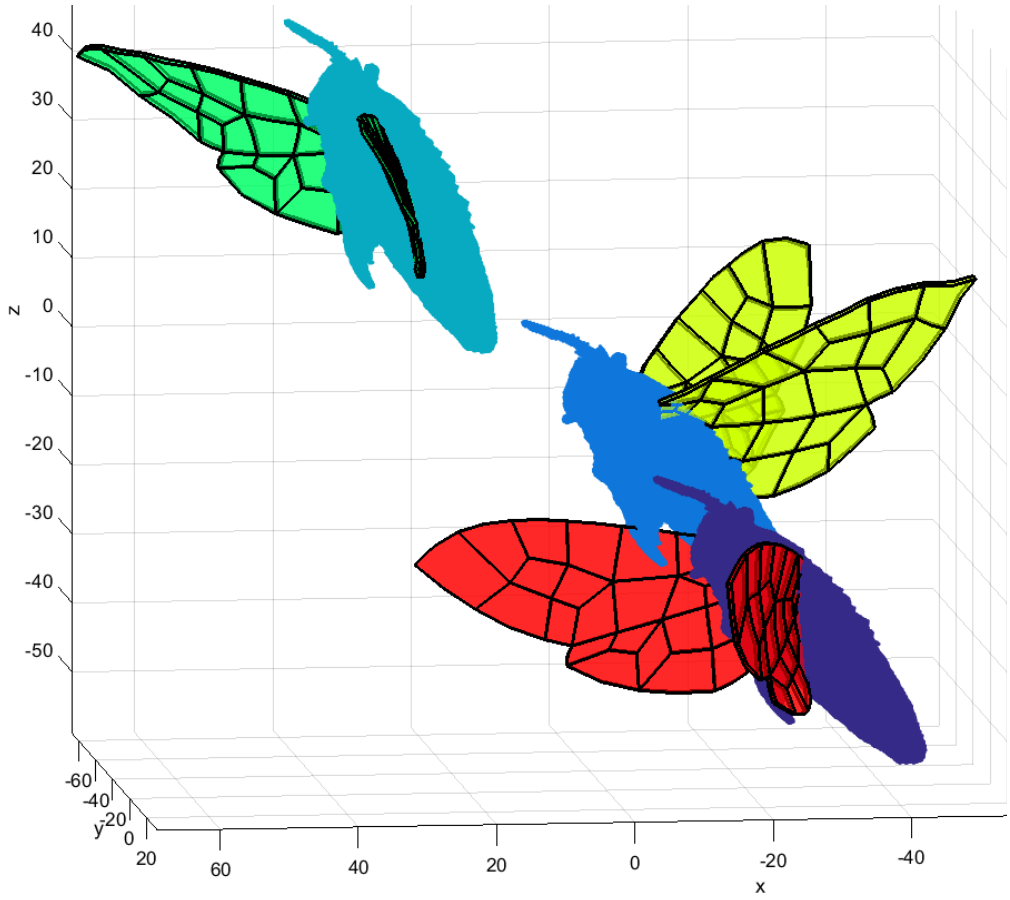
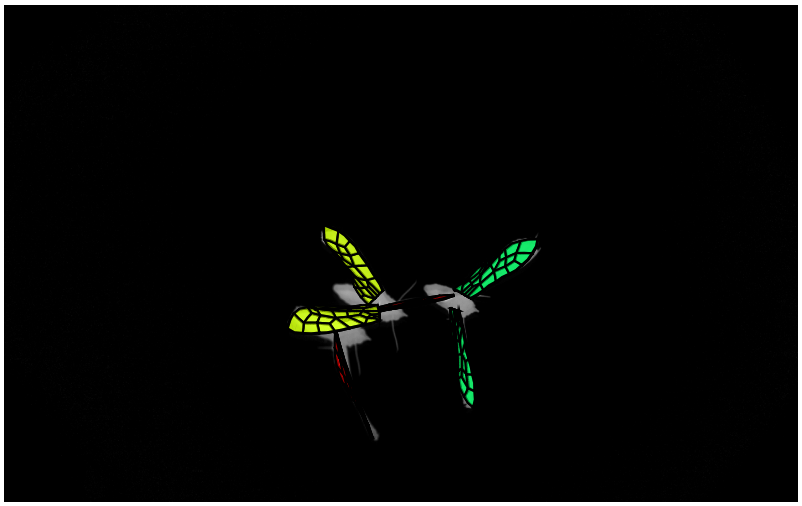
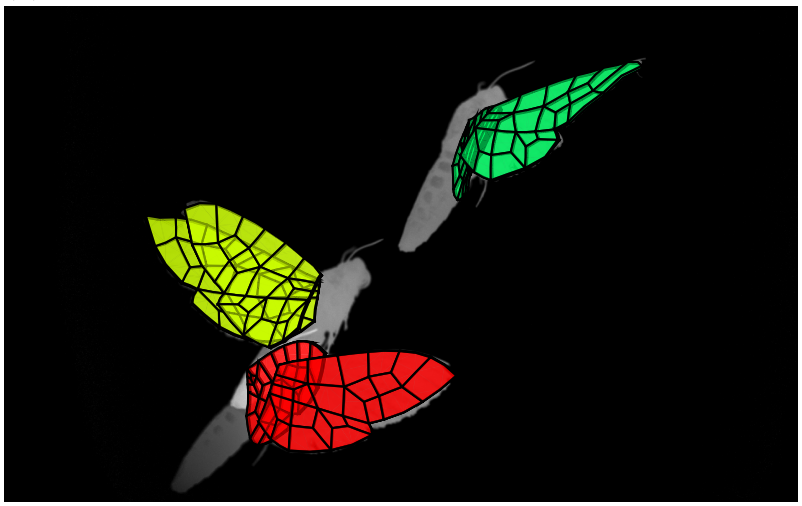


Figure 3.7: Reconstructed motion at three different time steps (mm).



(a) View from camera XY

(b) View from camera XZ



(c) View from camera YZ

Figure 3.8: Snapshots of the projections of motion estimation on the raw images.

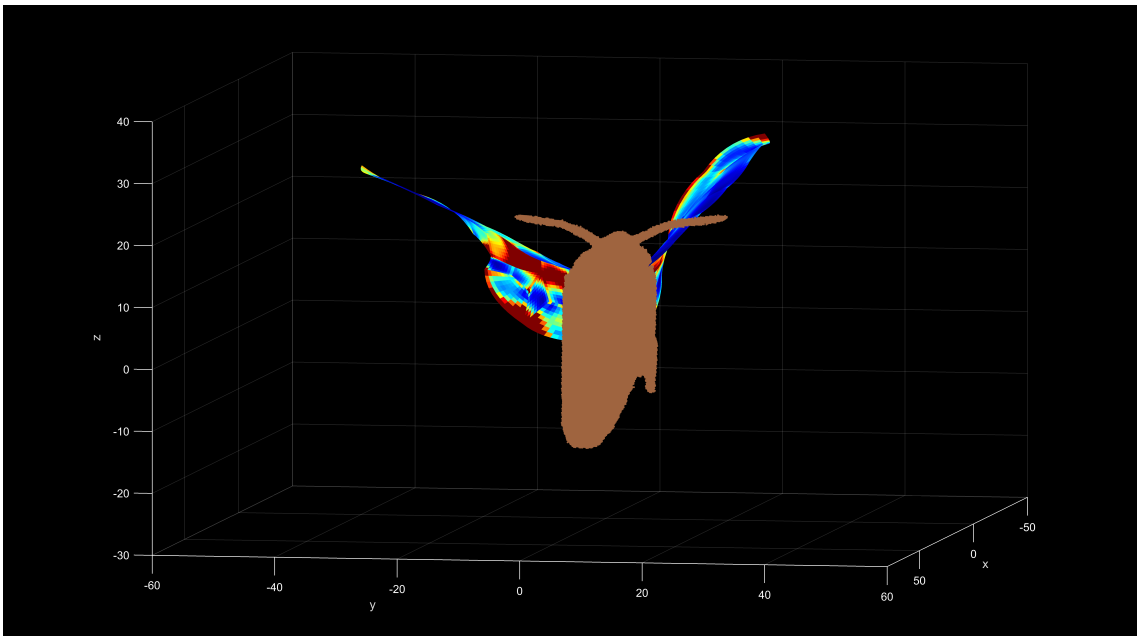


Figure 3.9: Residue of strain after optimization.

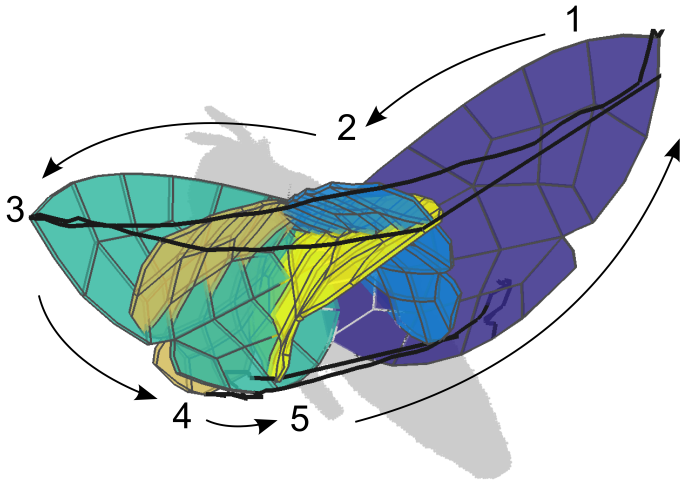
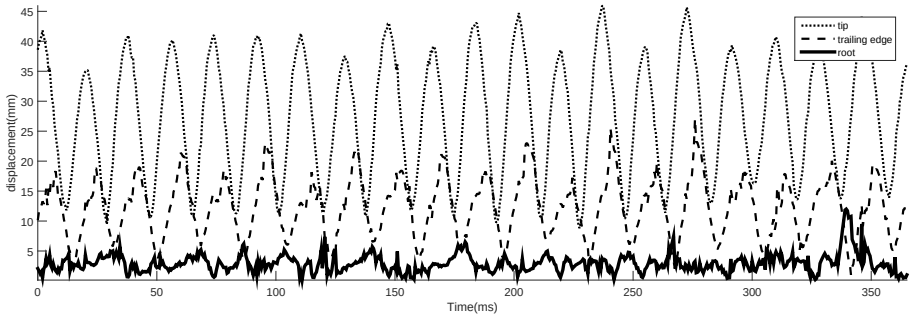
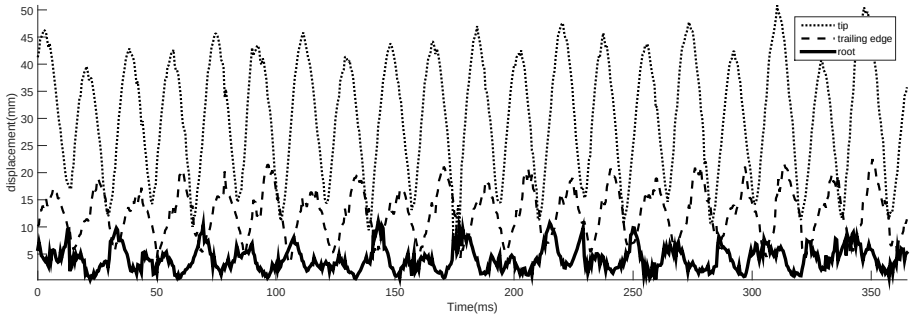


Figure 3.10: Multiple snapshots of the wing motion and the tip and the trailing edge trajectories in one cycle. The effective stroke plane is defined as the best planar fit of the tip and root trajectories over a cycle.



(a) Left wing plot



(b) Right wing plot

Figure 3.11: Amplitudes of oscillations of the left and right wings.

3.4.2 Proper orthogonal decomposition

Proper Orthogonal Decomposition (POD) is a convenient tool to extract the dominating operational modes in the least square sense[47]. Some introduction to POD can be found in Section A.5. The matrix $\mathbf{A}$ of measured ensemble of the tuples of nodal displacement of FE wings in consecutive time steps is assembled as.

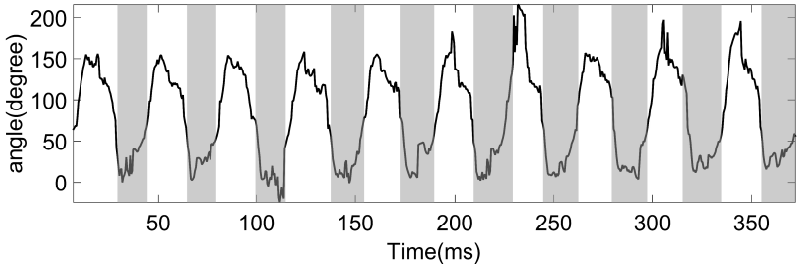
$$\mathbf{A} = \left[{}^{w}\mathbf{q}(t_1) \mid {}^{w}\mathbf{q}(t_2) \cdots {}^{w}\mathbf{q}(t_n) \right] \tag{3.25}$$

The POD modes are obtained by solving singular value decomposition problem of the matrix $\mathbf{A}$. In order to decouple the mode shapes from rigid-body motion, the deformation of the wing is defined in W-frame. The result results obtained for the POD mode shapes are shown in Figure 3.17. Chord-wise bending motion is observed in the first mode. Torsional motion is also seen in the first mode when one examines the constant displacement contour line drawn in black. The second mode is bending motion in span-wise direction, and the third mode is insignificant. The percentage of the singular value of each mode is shown in Figure 3.18.

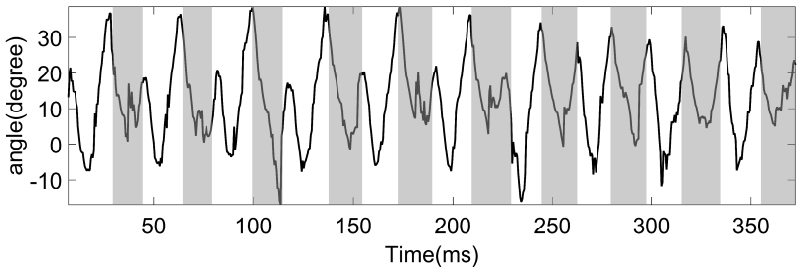
It is noted that the sequence of the dominant mode shapes in flapping motion is different from the sequence observed observed during modal analysis experiments. From Table 3.1, the ratio of the linear natural frequency of each mode to the observed flapping frequency obtained from the FFTs of the Euler angles of the root motion is listed.

Comparing the time history of the intensity of the first POD mode to that of

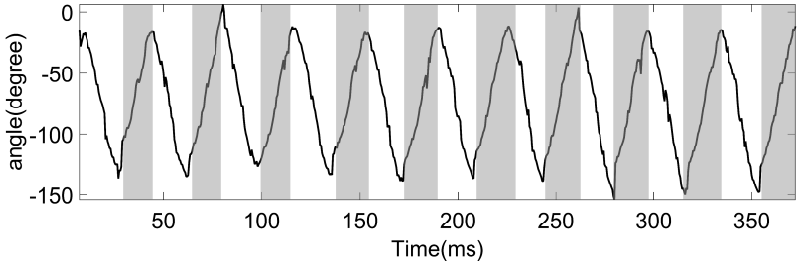
the rigid-body motion, the deformation reaches the extremum quickly and oscillates around the extremum before jumping to the opposite direction.



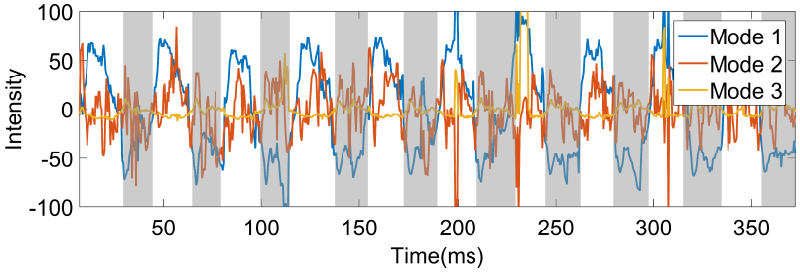
(a) Time history of ρ



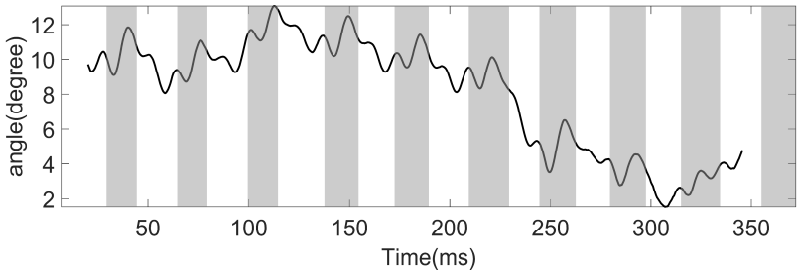
(b) Time history of ϕ



(c) Time history of θ

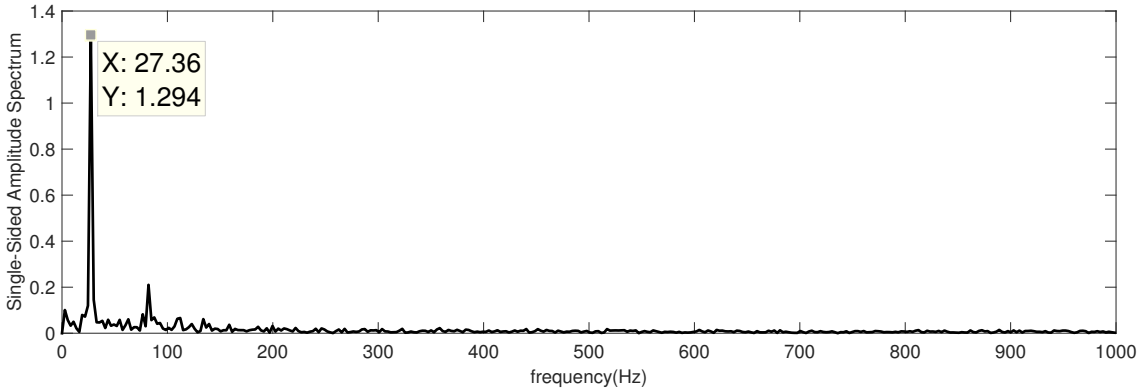


(d) Time history of the proper orthogonal decomposition modes

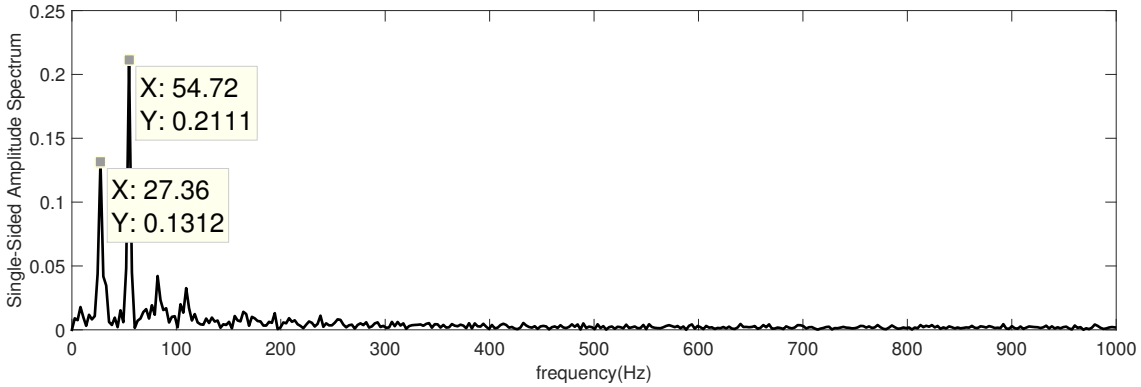


(e) Effective stroke angle

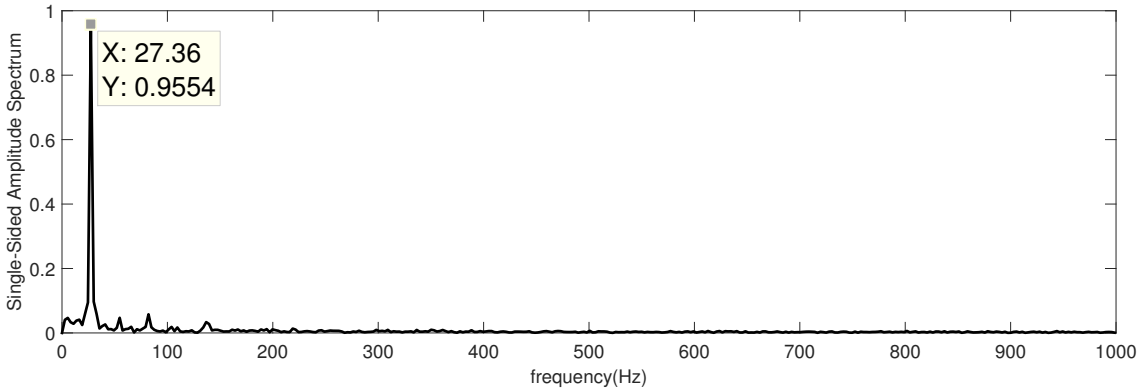
Figure 3.12: Time histories for rigid-body motion of the root in R' frame the proper orthogonal decomposition modes in W-frame, and the effective stroke angle of the right wing.



(a) ρ

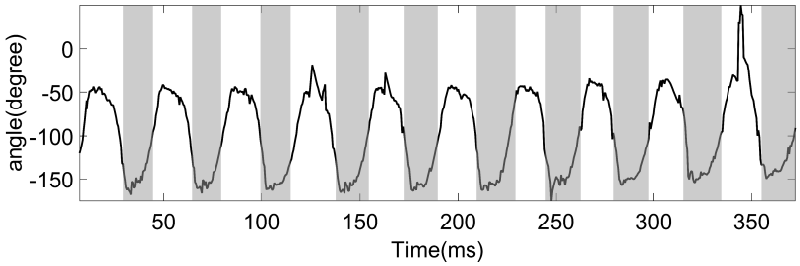


(b) ϕ

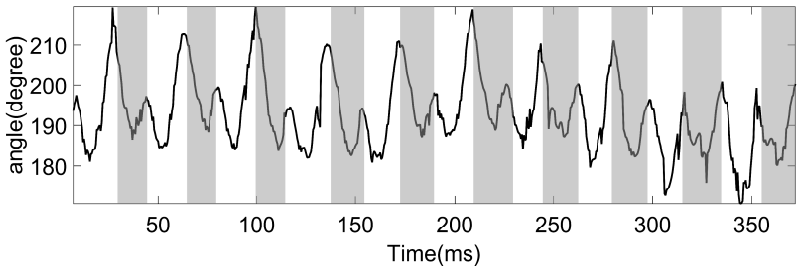


(c) θ

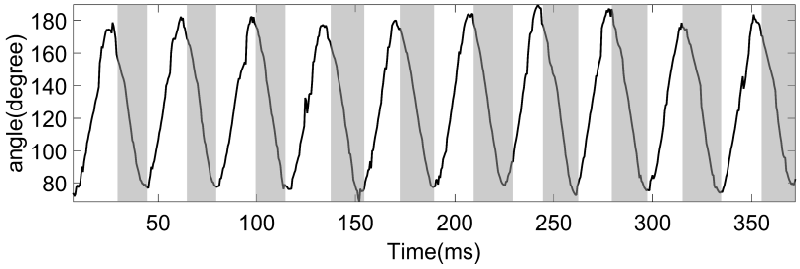
Figure 3.13: Fast Fourier transform results for the right root Euler angles.



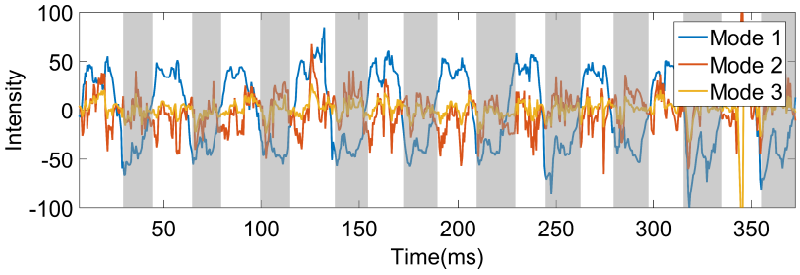
(a) Time history of ρ



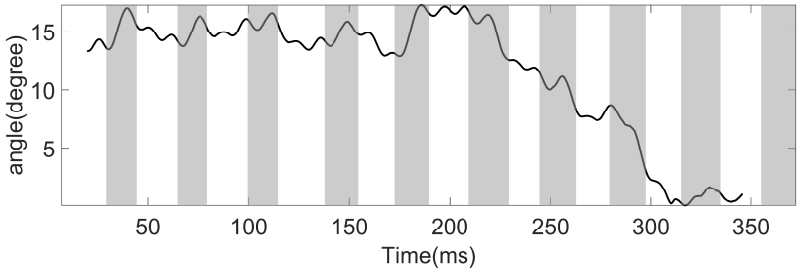
(b) Time history of ϕ



(c) Time history of θ

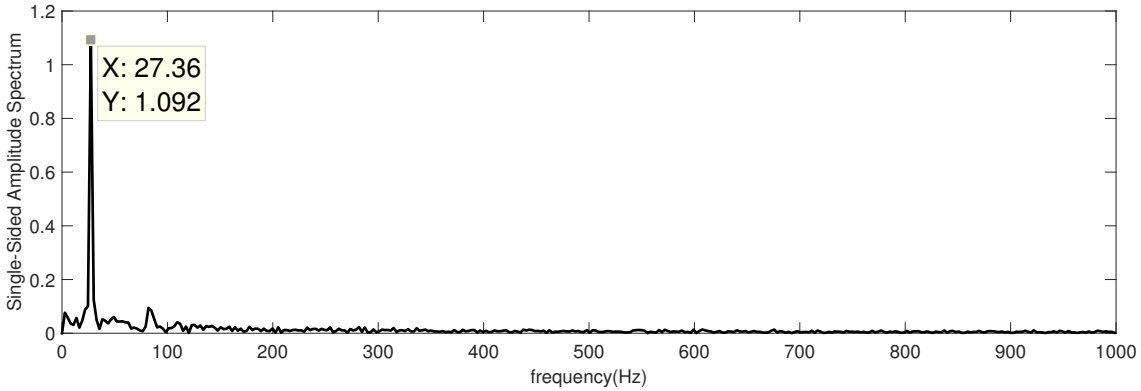


(d) Time history of proper orthogonal decomposition modes

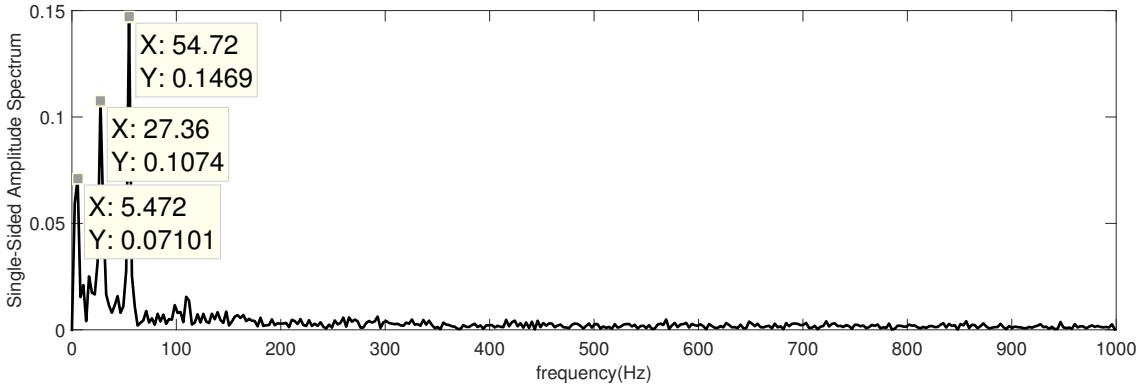


(e) Effective stroke angle

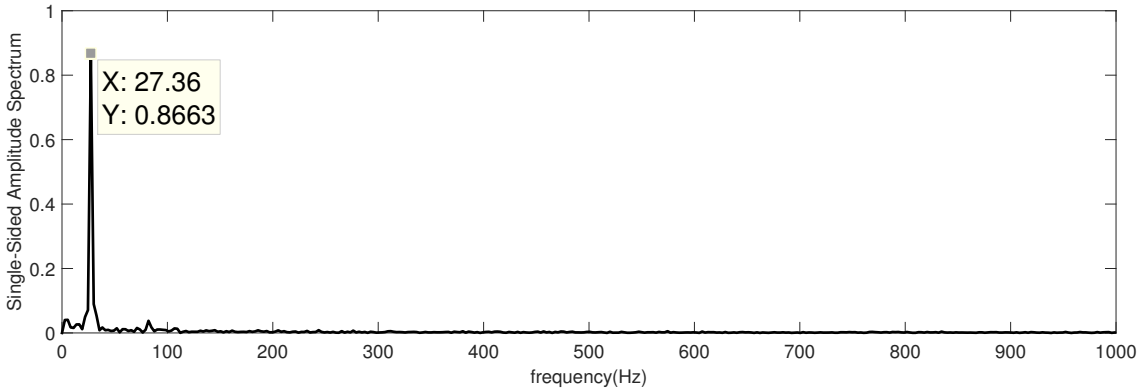
Figure 3.14: Time histories for rigid-body motions of the root in R' frame, the proper orthogonal decomposition modes in W-frame, and the effective stroke angle of the left wing.



(a) ρ

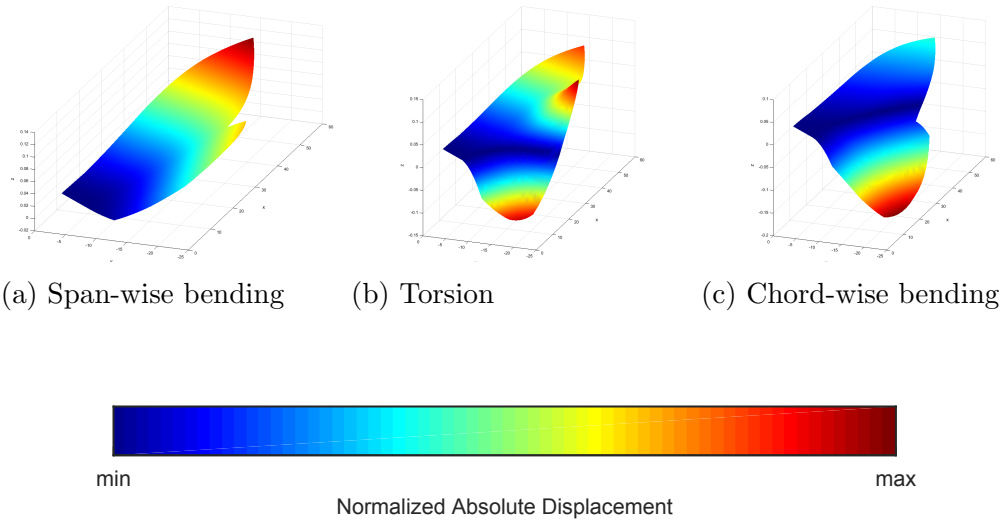


(b) ϕ



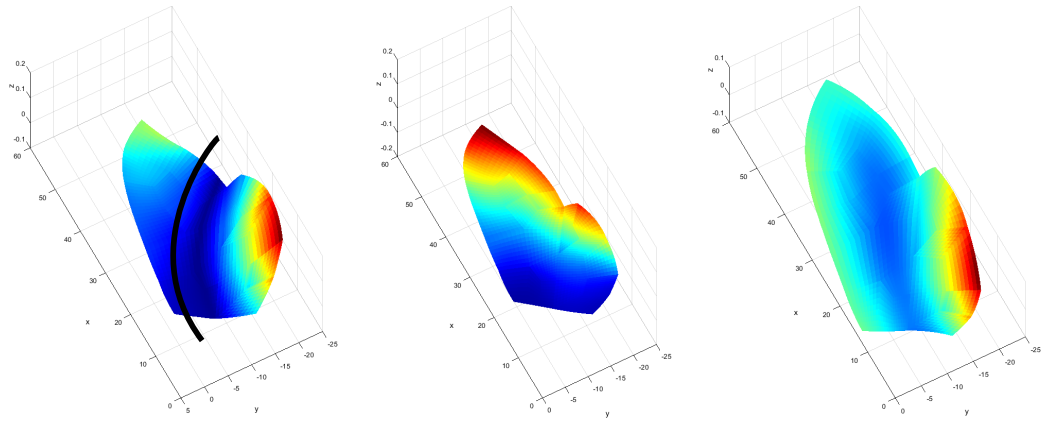
(c) θ

Figure 3.15: Fast Fourier transform results for the left root Euler angles.



(a) Span-wise bending (b) Torsion (c) Chord-wise bending

Figure 3.16: Linear mode shapes of right wing.



(a) First mode (b) Second mode (c) Third mode

Figure 3.17: Proper orthogonal decomposition mode shapes of right wing. Black line shows the contour of constant displacement.

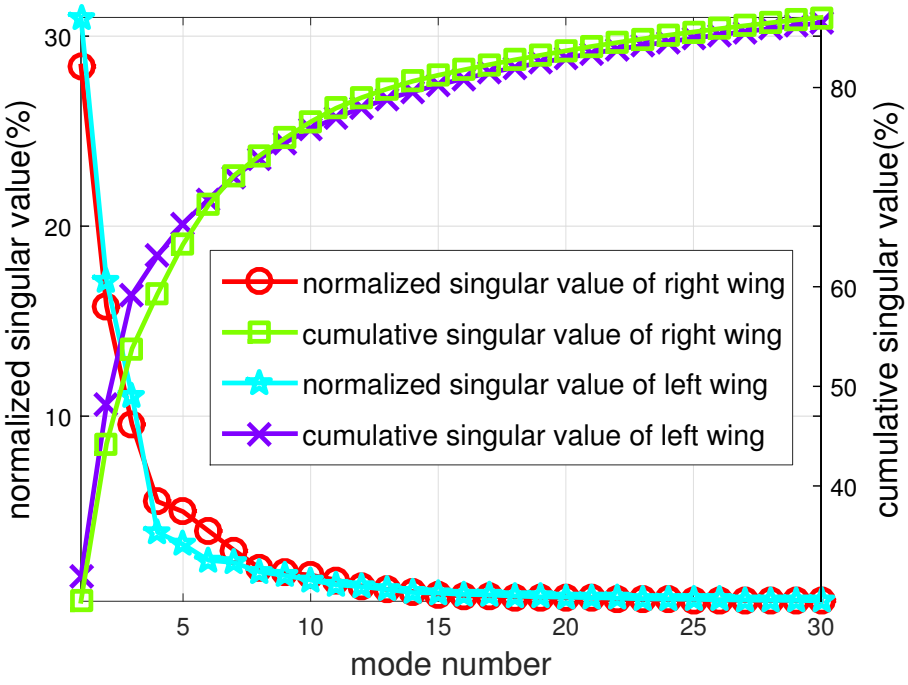


Figure 3.18: Percentage and cumulative values of the singular values.

Chapter 4: Integration of experimental data into fluid-body simulation

In this chapter, the aerodynamic performance of a hawkmoth in free-flight motion discussed in the previous chapter is studied. The adopted fluid model is the unsteady vortex-lattice method (UVLM). UVLM has been known for its appealing compromise between the computational cost and fidelity[49, 50, 51] , and its capability has been extended studies of flapping insects. Roccia et al. [14] have compared the aerodynamic load of a flapping wing to experimental data of a robo-fly under flapping motion. With the encouraging results in this recent study, the author and the coworkers here explore fluid-body simulations further by integrating the rotating and deforming motion of the hawkmoth estimated from the previous chapter with UVLM.

In the following sections, a short description of the UVLM is presented. Next, the motion is parameterized in both spatial and time domains to obtain continuous description of position and velocity at an arbitrary time instant. Modification made from the extension of modeling the leading edge separation in reference [14] is described. Finally, preliminary results of the wake kinematic is visualized, and the time history of the aerodynamic load over ten wing beat cycles is estimated cycle

by cycle.

4.1 Fluid model: unsteady vortex-lattice method

UVLM is used to solve the Navier-Stokes equations, when the flow is modeled as invicid potential flow. As a consequence, the boundary layer becomes infinitesimally thin, and the Reynold's number goes to infinity. The particles on the lifting surface as well as the wake are convected from the sharp edges. The diffusion effect of the rotational particles is neglected, and therefore, the conservation of circulation, or namely, Kelvin's theorem, is satisfied.

The boundary layer, or the bound-vortex sheet is discretized as a lattice of vortex segments of constant circulation. The vorticity field of the flow is solved from the no-penetration boundary conditions at the control points of the lifting surface, where the velocity field is calculated by the superposition of the induced velocity from each vortex segment by using Biot-Savart Law. The vortex segments at the sharp edges are convected with local particle velocity in order to satisfy the unsteady Kutta condition. The wake is thus formed and the circulation of each vortex segment is conserved in accordance with Kelvin's theorem.

The aerodynamic load on each element of lifting surface is approximated by calculating the pressure jump across the vortex sheet. The pressure jump is determined from the unsteady Bernoulli's Equation, which is derived from the Navier-Stokes equations , after assuming that the flow is irrotational and invicid. Note that this step is independent of the calculation of kinematics of the flow. The reader is

directed to Preidikman's work[50] for more details.

4.2 Prescribed motion

The free-flight motion of *Manduca sexta* estimated from Chapter 3 is imported here as the prescribed motion of the lifting surfaces. Since, the spatial expression of the *Manduca sexta* wings is continuous, the author can discretize the wing surfaces further down to any desired characteristic length from the original quadratic elements to linear elements. In order to obtain the displacement and velocity profile of the motion at any time step, the time sequence of the displacement field estimated from flight data is decomposed into non-periodic signal and near-periodic signal, and the signals are curved-fitted by using polynomials and FFT information.

Let the position vector of one point on the wing in R' frame ${}^{R'}\mathbf{r}$ estimated at time t_k be expressed as

$$ {}^{R'}\mathbf{r}(t_k) = \mathbf{R}(t_k)\,{}^{w}\mathbf{r}(t_k) + \mathbf{d}(t_k) \tag{4.1} $$

where $\mathbf{R}(t_k)$ and $\mathbf{d}(t_k)$ is the transformation from the global frame to W-frame.

The displacement in W-frame ${}^{w}\mathbf{r}$ is considered local deformation that is further decomposed by POD analysis. The displacement of the W-frame $\mathbf{d}(t_k)$ is decom-

posed into the combination of body motion and the motion relative to the B-frame.

$$\begin{aligned} {}^{R'}\mathbf{r}(t_k) =& \mathbf{R}(t_k){}^{w}\mathbf{r}(t_k) + \mathbf{d}^{B/O}(t_k) + \mathbf{d}^{W/B}(t_k) \\ =& \mathbf{R}(t_k)\mathbf{U}({}^{w}\mathbf{r})\mathbf{\Lambda V}(t_k) + \mathbf{d}^{B/O}(t_k) + \mathbf{d}^{W/B}(t_k) \end{aligned} \tag{4.2}$$

$\mathbf{d}^{B/O}$ is curve-fitted with spline interpolation.

The velocity is then

$${}^{R'}\dot{\mathbf{r}}(t_k) = \mathbf{R}(t_k)\left({}^{w}\dot{\mathbf{r}}(t_k)\right) + \dot{\mathbf{R}}(t_k)\left({}^{w}\mathbf{r}(t_k)\right) + \dot{\mathbf{d}}^{B/O}(t_k) + \dot{\mathbf{d}}^{W/B}(t_k) \tag{4.3}$$

From equation (A.1), the time derivative of $\mathbf{R}$ is

$$\dot{\mathbf{R}} = \dot{\mathbf{R}}_\theta\mathbf{R}_\phi\mathbf{R}_\rho + \mathbf{R}_\theta\dot{\mathbf{R}}_\phi\mathbf{R}_\rho + \mathbf{R}_\theta\mathbf{R}_\phi\dot{\mathbf{R}}_\rho \tag{4.4}$$

FFTs are then applied to the Euler angles, and $\mathbf{d}^{W/B}$. A minimum number of frequency terms are taken to reconstruct the analytic expression in time domain. The residues are considered as deformation, and SVD is performed. Again, with the spatial deformation described in $\mathbf{U}$ from equation (A.55), the displacement field at any position can be evaluated. Each column of $\mathbf{V}$ contains the information of unitary time history of intensity of the motion. FFT is applied to this information. The deformation of the new mesh for UVLM at time t becomes

$$\mathbf{U}_{UVLM}\mathbf{\Lambda V}(t) \tag{4.5}$$

4.3 Effective angle of attack

Whether the vortex shedding from the leading edge is happening or not depends on the angle between the local fluid velocity and the wing panel. Following Roccia et al.'s work[14], an on/off mechanism on each panel at the leading edge in included.

The local fluid velocity at the edge of one element can be calculated by summing up all the induced velocity over all of the vortex segments. Substracting out the component parallel to the edge direction, the effective angle of attack can be calculated from the angle between the element normal and the fluid velocity perpendicular to the edge, as shown in Figure 4.1. The component of local fluid velocity perpendicular to the edge is

$$\boldsymbol{V}_{attack} = \boldsymbol{V}_{fluid} - (\boldsymbol{V}_{fluid} \cdot \boldsymbol{L}_2)\boldsymbol{L}_2 \tag{4.6}$$

which leads to

$$\cos(\gamma) = \left| \frac{\boldsymbol{V}_{attack} \cdot \boldsymbol{n}}{\|\boldsymbol{V}_{attack}\|} \right| \tag{4.7}$$

The vortex ring is convected if the effective angle of attack is greater than a threshold value α_c

$$\alpha = \frac{\pi}{2} - \gamma > \alpha_c \tag{4.8}$$

or equivalently,

$$\left| \frac{\boldsymbol{V}_{attack} \cdot \boldsymbol{n}}{\|\boldsymbol{V}_{attack}\|} \right| > \cos(\frac{\pi}{2} - \alpha_c) \tag{4.9}$$

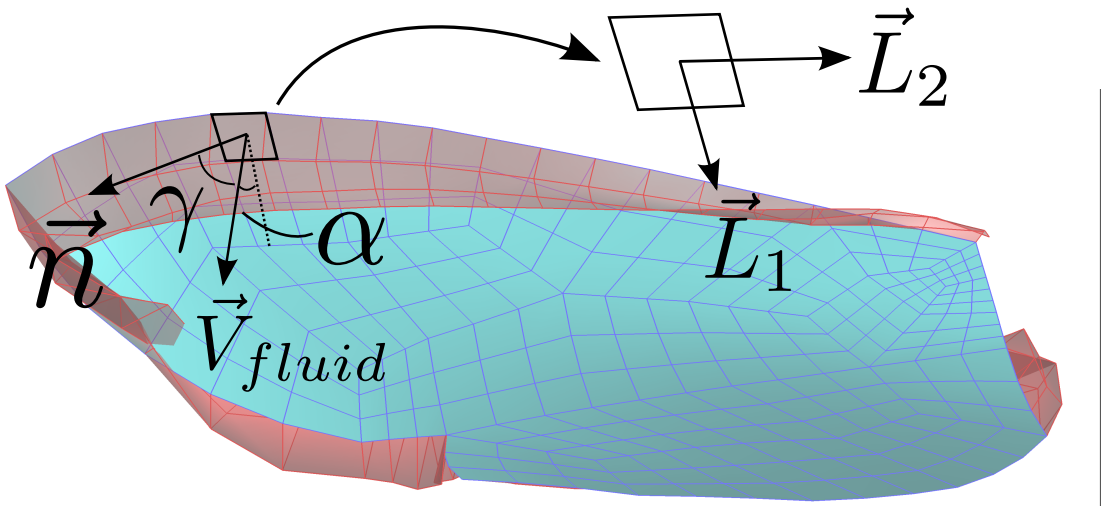


Figure 4.1: Effective angle of attack (AoA) of one element.

4.4 Preliminary results

The simulation is carried out with one cycle of the wing beat starting at different time steps. The vortex sheet of the wing is generated from the FE mesh, discretized further with the characteristic length of 1.5 millimeters. The characteristic velocity is chosen as the mean velocity over one cycle. The distribution of the length of each vortex segment and the velocity at each center of element over time is shown in **??**. This simulation is carried out under the pure translational B"-frame; therefore the free-stream velocity is the negative mean body velocity.

The mesh information is listed in Table 4.1. In Figure 4.2 and **??**, the amount of how the length and velocity of each element varying through time are shown in order to justify the choice of the nondimensional variables. The y-axis of each plot represents the element number from the left wing, the right wing, and the body

mesh, respectively.

The simulations of the wake visualization of a representative snapshots of the downsroke and upstroke are shown in Figure 4.4 and Figure 4.5, respectively. In each graph, the mesh of the hawkmoth and the wake are plotted in blue and red, and the circulation calculated from UVLM is shown as contour plot as well. Strong tip vortex and trailing edge vortex are observed. The leading edge vortex is convected outward to the spanwise direction and integrated into the tip vortex.

In Figure 4.6, the estimated aerodynamic load and power applied on the moth with initial condition at each downstroke and upstroke are shown. It can be seen that the estimated aerodynamic load obtained from the simulations does not vary drastically when started from an upstroke versus a downstroke. This indicates that the impulse start simulation provides satisfactory results without consideration of the past strokes of the flight motion. The force acting on the hawkmoth in x-direction changes sign between the back and forth motion of the wings. The area formed by the force-time plot represents the change of linear momentum of the hawkmoth, and the area formed during the upstroke is slightly larger than the downstroke. Same result is revealed when one recalls the estimated body motion shown in Figure 2.12, wherein the geometric center of the hawkmoth traveled forward.

There is a peak lift force during each upstroke period and downstroke period. From Figure 3.12, the effective stroke angle, which represents the angle between the best planar fit of the tip and root motion and the horizon, is approximately 10 degrees. However, due to the flexibility of the wing, the remaining part of the wing is positioned at a large angle of attack with respect to the flow, as shown in

Characteristic length	: 1.5 mm
Characteristic velocity	: 1500 mm/s
Characteristic time	: 1 ms
Number of elements	: 1136

Table 4.1: Information for the UVLM simulation.

Figure 3.10.

4.5 Discussions

In this chapter, the estimated motion of a free-flying hawkmoth is integrated into a fluid-body simulation environment in order to gain more information from the natural flapping flyers. The interactions between vortices are visualized, and the aerodynamic load is estimated. Since the estimation of the aerodynamic load does not vary drastically regarding with the initial condition as well as the history of the past cycle, UVLM can be a powerful computationally efficient tool to carry on further investigation such as fluid-structure interaction simulation or parametric studies.

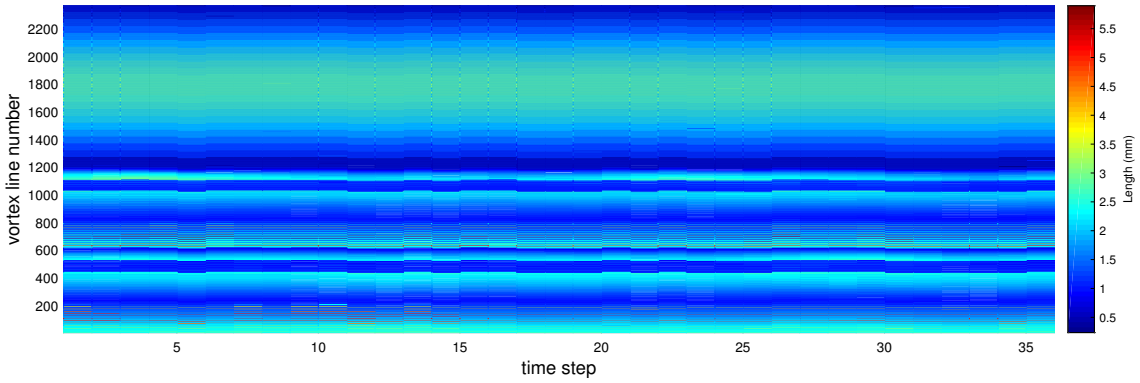


Figure 4.2: Length distribution over the wings and body.

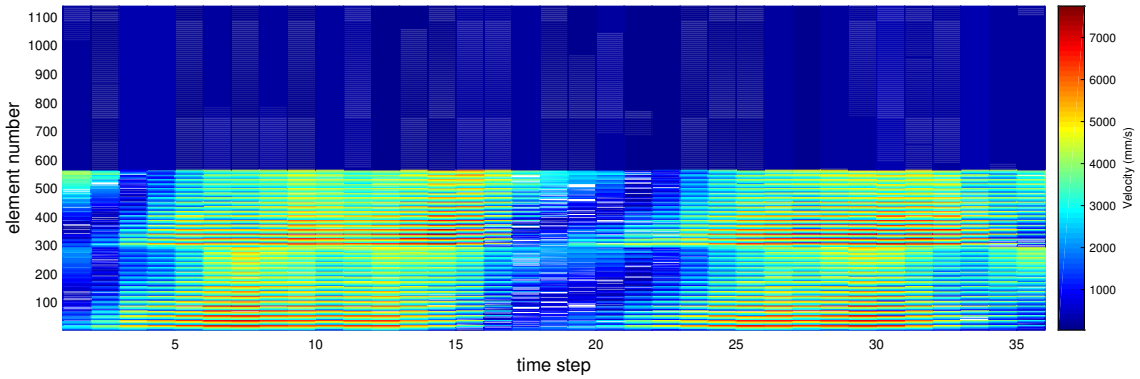
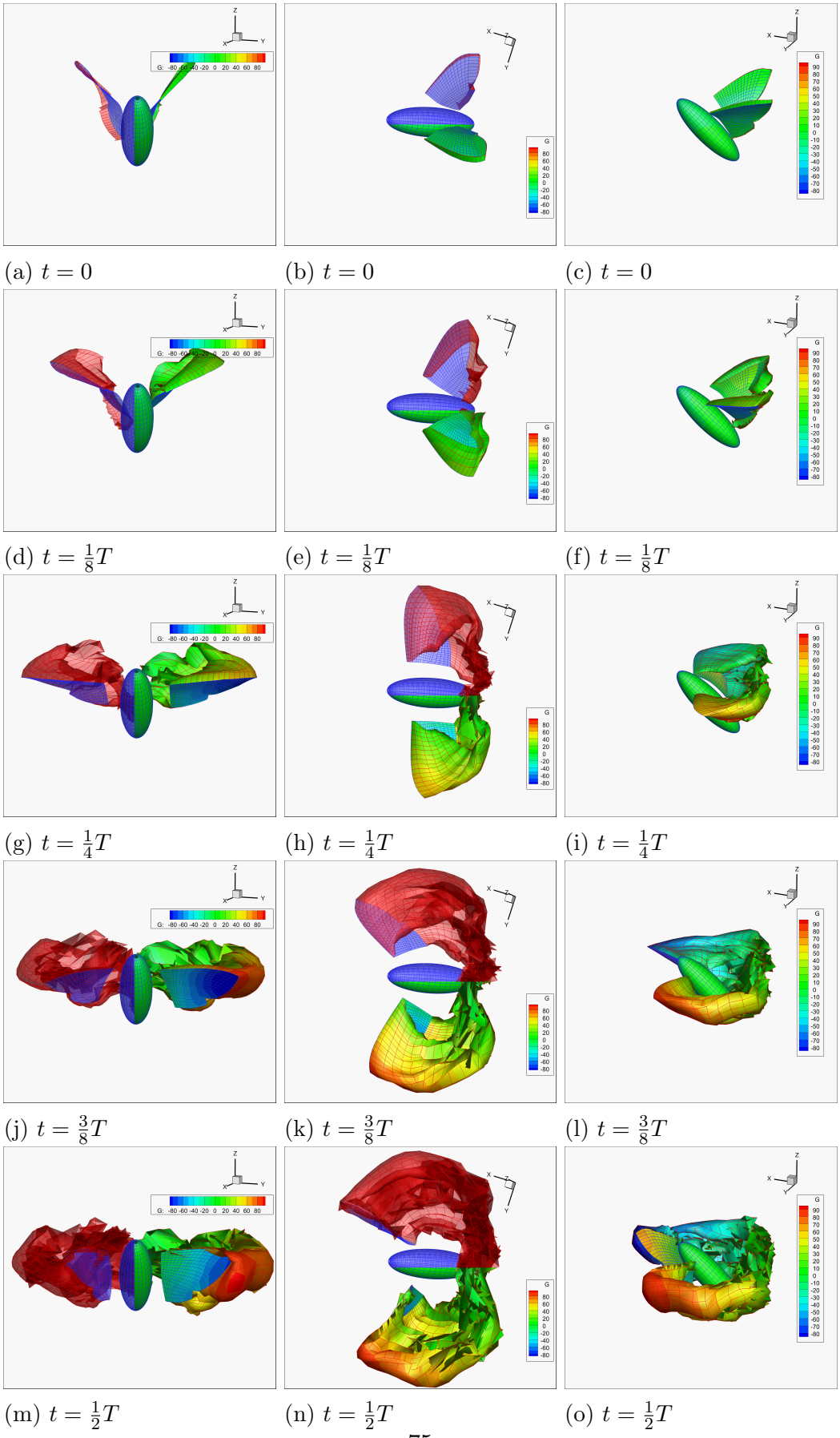


Figure 4.3: Velocity distribution over the wings and body.



(a) $t = 0$ (b) $t = 0$ (c) $t = 0$

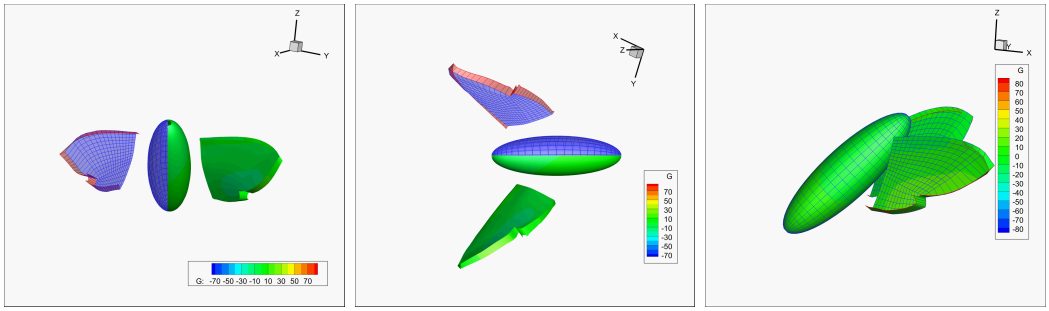
(d) $t = \frac{1}{8}T$ (e) $t = \frac{1}{8}T$ (f) $t = \frac{1}{8}T$

(g) $t = \frac{1}{4}T$ (h) $t = \frac{1}{4}T$ (i) $t = \frac{1}{4}T$

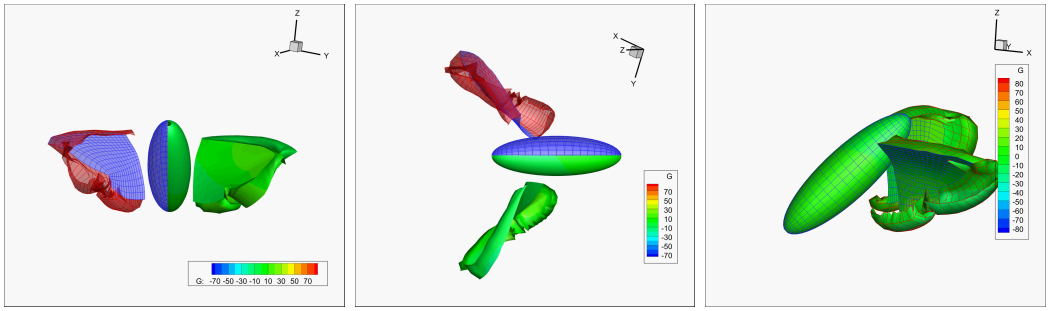
(j) $t = \frac{3}{8}T$ (k) $t = \frac{3}{8}T$ (l) $t = \frac{3}{8}T$

(m) $t = \frac{1}{2}T$ (n) $t = \frac{1}{2}T$ (o) $t = \frac{1}{2}T$

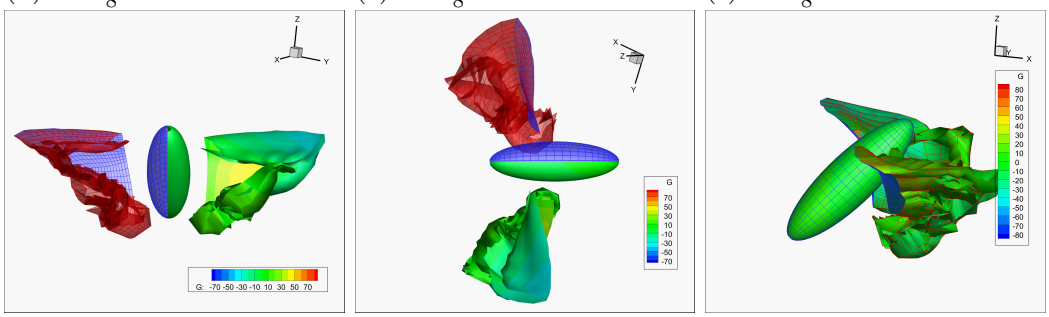
Figure 4.4: Wake pattern(red mesh) and the strength of circulation(G) of down-stroke.



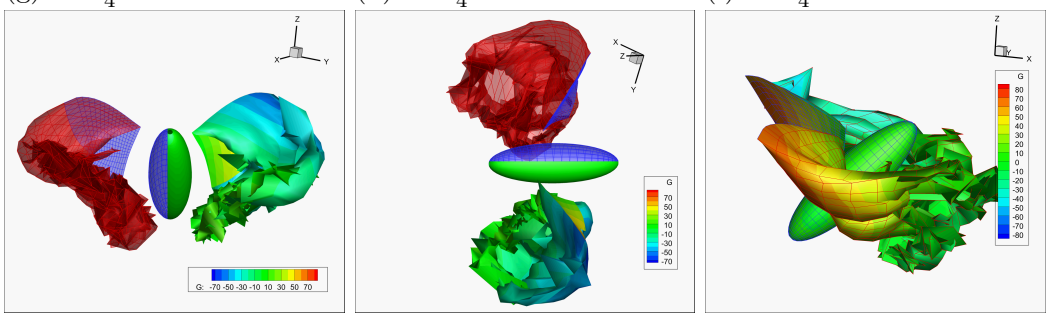
(a) $t = 0$ (b) $t = 0$ (c) $t = 0$



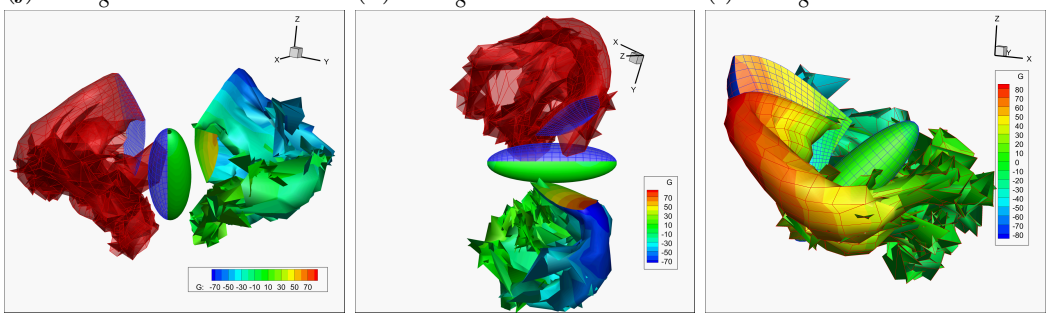
(d) $t = \frac{1}{8}T$ (e) $t = \frac{1}{8}T$ (f) $t = \frac{1}{8}T$



(g) $t = \frac{1}{4}T$ (h) $t = \frac{1}{4}T$ (i) $t = \frac{1}{4}T$

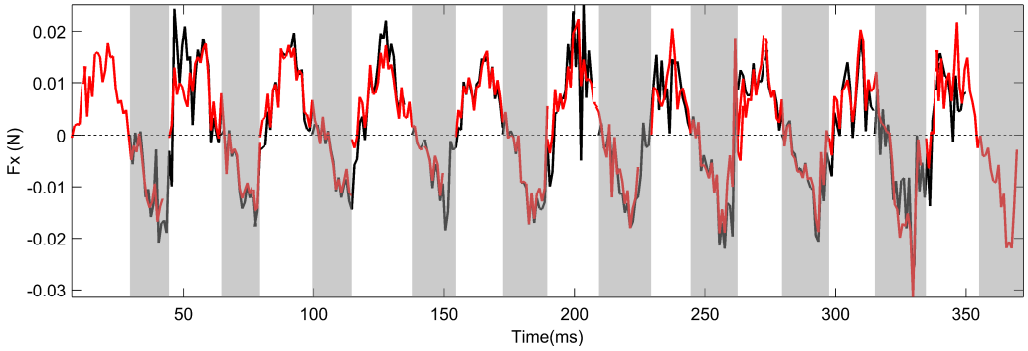


(j) $t = \frac{3}{8}T$ (k) $t = \frac{3}{8}T$ (l) $t = \frac{3}{8}T$

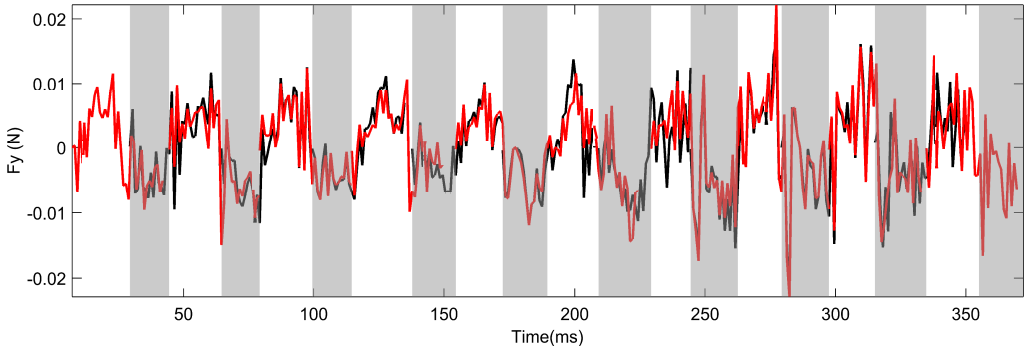


(m) $t = \frac{1}{2}T$ (n) $t = \frac{1}{2}T$ (o) $t = \frac{1}{2}T$

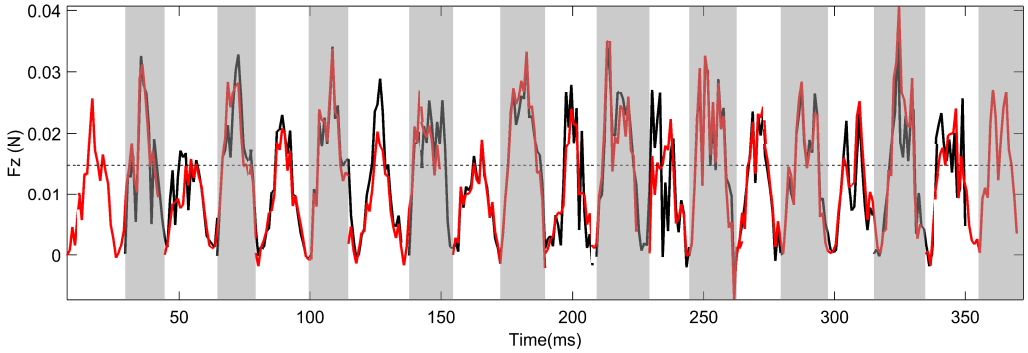
Figure 4.5: Wake pattern(red mesh) and the strength of circulation(G) of upstroke.



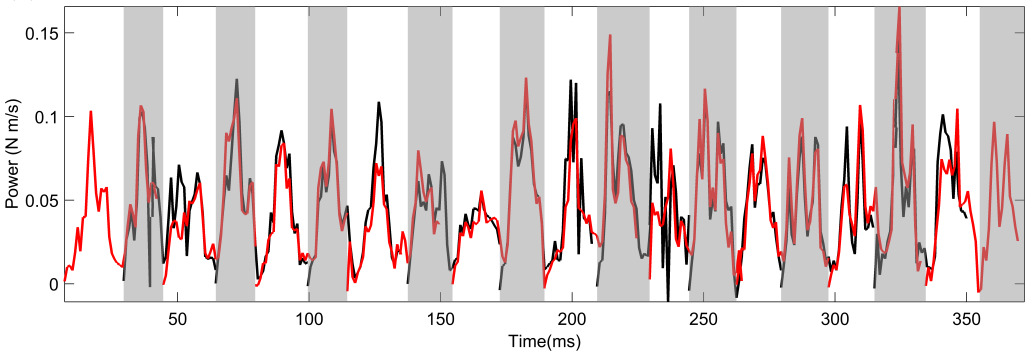
(a) F_x plot



(b) F_y plot



(c) F_z plot



(d) *Power* plot

Figure 4.6: Time histories of the aerodynamic load and power. Motion following an impulse start in a downstroke is shown using black lines, and motion following an impulse start in a upstroke is shown in red lines.

Chapter 5: Concluding remarks and future work

5.1 Summary

In this thesis, the author has proposed a non-invasive approach to estimate the motion of a 3D deformable body from multiple angles of 2D snapshots. Steps are taken toward integration of experimental data into fluid-structure simulations.

In Section 3.3, the author showed that the estimated configuration from 2D snapshots is close to the true configuration. This approach is then applied to free flight data of a hawkmoth. The estimated motion may be of interest to biologists and engineers interested in the mechanisms of the flapping flight motion. Although this approach is applied to reconstruct the free flight wing motion of a hawkmoth, the tools and framework is applicable to any large motion visualization and estimation with multiple spatial sensors.

One nice feature of the experimental approach of this thesis is that entire process requires less manual work than most contemporary approaches. Once the experimental setup is calibrated, the flapping motion of other animals or mechanical systems may be reconstructed as shown in this thesis work.

5.2 Recommendations for future work

As the focus of this work has been the motion of a single moth flight, a battery of experiments of different moths and different animal types is further necessary to create a comprehensive understanding of the mechanism of flapping flight. Additionally, the development of the development of a fluid-body simulation with integrated experimental data can aid in discovering key components of flapping flight data, from different experimental tests.

Constructing a simulation environment for FSI analysis combining the hyperelastic FE model and UVLM can not only help gain a comprehensive understanding of flapping flight mechanisms, but also help develop a pragmatic tool to design an MAV.

Appendix A: Appendices

A.1 Frame definition

A.1.1 Coordinate systems

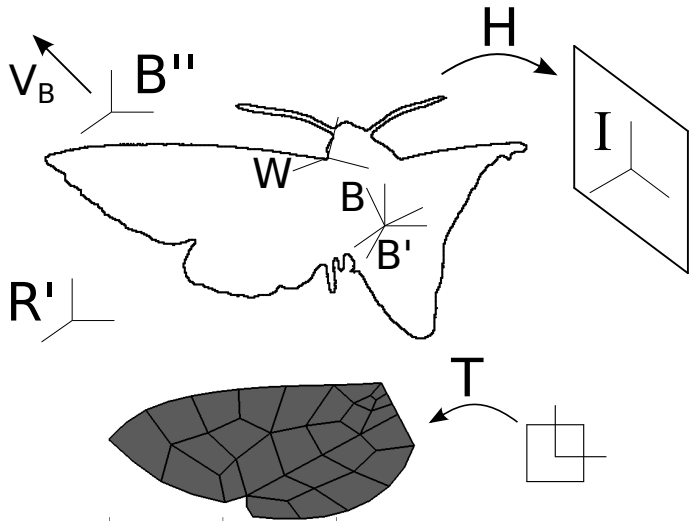


Figure A.1: Frame definition.

R'	:	Newtonian frame
W	:	rotational wing frame attached at the base of a wing
B	:	rotational body frame fixed at the geometric center of the body
B'	:	translational body frame fixed at the geometric center of the body
B"	:	translational body frame with constant velocity $\boldsymbol{V}_B$
I	:	image plane
H	:	mapping from R' to I plane
T	:	mapping from natural coordinates to the finite element representation in R'

A.1.2 Rigid body rotation

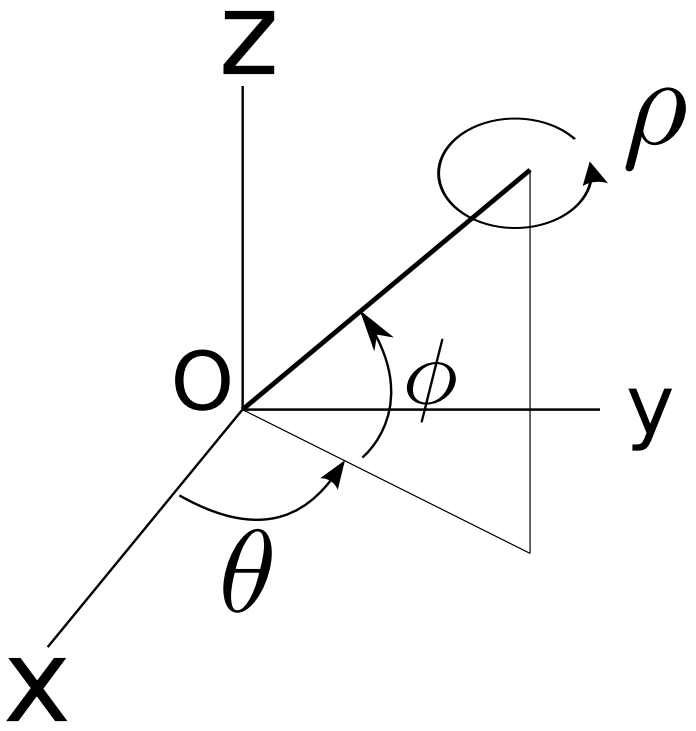


Figure A.2: Definition of the Euler angles in rigid body rotation.

The rotation matrix can be expressed by Euler angles. Here, this takes the form

$$\begin{aligned}\mathbf{R} =& \mathbf{R}_\theta \mathbf{R}_\phi \mathbf{R}_\rho \\ =& \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos(\phi) & 0 & -sin(\phi) \\ 0 & 1 & 0 \\ sin(\phi) & 0 & cos(\phi) \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\rho) & -sin(\rho) \\ 0 & sin(\rho) & cos(\rho) \end{bmatrix}\end{aligned} \tag{A.1}$$

A.2 Camera calibration

This section tries to convey the general idea of geometric camera calibration. It covers just enough concepts to implement calibration using the built-in functions in MATLAB, because part of the documentations in MATLAB is unclear, and the notations are different from the references pointed out by the documentation. For more details, the readers are directed to the studies of Hartley and Zisserman [2] and Heikkila and Silvén [52].

A.2.1 Direct linear transform

One can write down the general form of equation (2.4) as

$$
\begin{aligned}
w \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} &= \mathbf{P} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}
\end{aligned}
\tag{A.2}
$$

Consider one known point in the world frame (X_i, Y_i, Z_i) and the corresponding image location (u_i, v_i). After substituting into equation (A.2) and eliminating the scalar w lead to

$$
\begin{bmatrix} X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -u_iX_i & -u_iY_i & -u_iZ_i & -u_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -v_iX_i & -v_iY_i & -v_iZ_i & -v_i \end{bmatrix} \begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ p_{14} \\ p_{21} \\ p_{22} \\ p_{23} \\ p_{24} \\ p_{31} \\ p_{32} \\ p_{33} \\ p_{34} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{A.3}
$$

Each known point contributes 2 equations. Therefore, at least 6 calibration points are needed to solve for all unknowns. Generally, one uses more points to have more accuracy. Thanks to the power of high speed camera, the author could obtain numerous data points effortlessly. For n points, the system needed to be solved is

$$
\mathbf{A}\boldsymbol{X} = \mathbf{0} \tag{A.4}
$$

where $\mathbf{A}$ is a $2n$ by 12 matrix.
This is an over-determined problem. Thus, one is seeking for the solution that minimizes the square sum of residue $\boldsymbol{\epsilon}$, which is

$$
\boldsymbol{\epsilon} = (\mathbf{A}\boldsymbol{X}) \tag{A.5}
$$

Since $\mathbf{A}$ can be arbitrarily scaled, an additional constraint is added in order to seek for an unique solution. In some literature, it has been suggested to force the value of p_{34} be 1. It can be numerically unstable if p_{34} is actually close to zero. Here, the

author uses the constraint that

$$\boldsymbol{X}^T \boldsymbol{X} = 1 \tag{A.6}$$

and the solution of X becomes a principal direction of $\mathbf{A}$. The solution is given in Subsection A.5.1.

A.2.2 Radial distortion

Here, the author only considers radial distortion with only two coefficients. The difference of the reprojection error compared to 3-coefficient radial distortion plus the tangential distortion model is minor in the present case. The process of reprojection is as following: Consider a point P $(X, Y, Z)_{world}$ in the R'-frame. The coordinate of P in camera frame is

$$(X, Y, Z)_{cam} = [\mathbf{R} \mid \mathbf{t}] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}_{world} \tag{A.7}$$

Now, define the normalized projection

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} \frac{X_{cam}}{Z_{cam}} \\ \frac{Y_{cam}}{Z_{cam}} \end{bmatrix} \tag{A.8}$$

Let the radius r be

$$r = \sqrt{x_n^2 + y_n^2} \tag{A.9}$$

The distorted point is

$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = \begin{bmatrix} x_n \\ y_n \end{bmatrix} (1 + k_1 r^2 + k_2 r^4) \tag{A.10}$$

Finally, the distorted image point is

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix} \tag{A.11}$$

Note that the third element is redundant.

A.2.3 Calibration procedure

In the calibration process, the author has executed the *Camera Calibrator APP* in MATLAB. By using a checkerboard specified in the documentation, the author first calibrates the intrinsic parameters for every camera separately. The author then records a series of candidate frames that the checkerboard was clearly

captured in all cameras, and the best angles that contained the largest amount of corner points was adopted. The extrinsic matrices were first estimated by the built-in function *extrinsics* in MATLAB, and the result was used as the initial condition of the nonlinear optimization problem, which minimizes the reprojection error, as described as Subsection A.2.2. The mean reprojection level is usually at the sub-pixel level.

A.3 More on multiple camera views: two-view geometry

A.3.1 Mathematical preliminaries: projective geometry in two dimensions

A general equation of a line in two dimension can be written as

$$ax + by + c = 0 \tag{A.12}$$

One can collect the parameters of a line into a vector

$$\boldsymbol{l} = \begin{bmatrix} a & b & c \end{bmatrix}^T \tag{A.13}$$

Consider a point in 2D plane

$$\boldsymbol{x} = \begin{bmatrix} x & y & 1 \end{bmatrix}^T \tag{A.14}$$

One can see that the point lies on the line if

$$\boldsymbol{x} \cdot \boldsymbol{l} = 0 \tag{A.15}$$

Next, a line $\boldsymbol{l}$ can be defined as

$$\boldsymbol{l} = \boldsymbol{x}' \times \boldsymbol{x} \tag{A.16}$$

where $\boldsymbol{x}$ and $\boldsymbol{x}'$ are two points. Since $\boldsymbol{l}$ is perpendicular to both $\boldsymbol{x}$ and $\boldsymbol{x}'$ in three dimensional space,

$$\boldsymbol{l} \cdot \boldsymbol{x} = \boldsymbol{l} \cdot \boldsymbol{x}' = 0 \tag{A.17}$$

So, the line $\boldsymbol{l}$ is formed by two points: $\boldsymbol{x}$ and $\boldsymbol{x}'$.

A.3.2 Epipolar geometry between two views

Consider a two view geometry as shown in Figure A.3. Suppose one is given the location of one image point $\boldsymbol{x}$ and two calibrated cameras. The point $\boldsymbol{X}$ is the back-projection point of $\boldsymbol{x}$ in three dimensional space. Assume that one knows two points on the back-projection ray:

1. Camera centre C, where $\mathbf{P}\boldsymbol{C} = 0$.

2. point $\mathbf{P}^+\boldsymbol{x}$, where $\mathbf{P}^+$ is the pseudo-inverse of $\mathbf{P}$.

$$\mathbf{P}^+ = \mathbf{P}^T(\mathbf{P}\mathbf{P}^T)^{-1} \tag{A.18}$$

It can be seen that the projection of $\mathbf{P}^+\boldsymbol{x}$ is such that

$$\mathbf{P}(\mathbf{P}^+\boldsymbol{x}) = \mathbf{I}\boldsymbol{x} = \boldsymbol{x} \tag{A.19}$$

Therefore, the ray can be expressed as

$$\boldsymbol{X}(\lambda) = \mathbf{P}^+\boldsymbol{x} + \lambda\boldsymbol{C} \tag{A.20}$$

While $\lambda = 0$, $\boldsymbol{X}(\lambda) = \mathbf{P}^+\boldsymbol{x}$, and $\lambda \to \infty$, $\boldsymbol{X}(\lambda) = \boldsymbol{C}$. From equation (A.17), the line $\boldsymbol{l}'$ joining $\boldsymbol{x}'$ and the projection of the point $\boldsymbol{C}$ to the other camera is

$$\begin{aligned} \boldsymbol{l}' =& (\mathbf{P}'\boldsymbol{C}) \times \mathbf{P}'(\mathbf{P}^+\boldsymbol{x} + \lambda\boldsymbol{C}) \\ =& (\mathbf{P}'\boldsymbol{C}) \times (\mathbf{P}'\mathbf{P}^+\boldsymbol{x}) \\ =& \mathbf{F}\boldsymbol{x} \end{aligned} \tag{A.21}$$

$\mathbf{P}'$ is the camera matrix for the other view. $\mathbf{F}$ is called the fundamental matrix. An important property of the fundamental matrix is that

$$\boldsymbol{x}'^T\mathbf{F}\boldsymbol{x} = \boldsymbol{x}'^T\boldsymbol{l}' = 0 \tag{A.22}$$

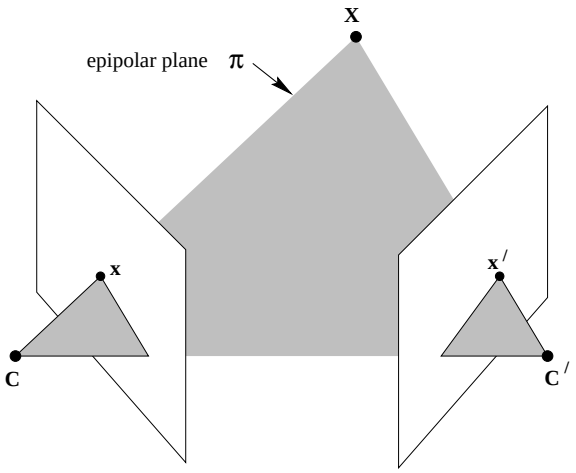


Figure A.3: Figure for epipolar geometry [2].

A.4 Calculations of gradients of the cost for dual optimization

A.4.1 Gradient of distance penalty

The distance d_j^2 at the minimum point can be written as

$$d_j^2 = (\mathbf{p} - \bar{\mathbf{r}}) \cdot (\mathbf{p} - \bar{\mathbf{r}}) \tag{A.23}$$

where $\bar{\mathbf{r}} = \mathbf{N}(\bar{\xi},\, \bar{\eta}) \cdot (\mathbf{x} + \mathbf{q})$. Computing the first term of equation (3.16) leads to

$$\begin{aligned} \frac{\partial d_j^2}{\partial q_k} &= \frac{\partial}{\partial q_k} (\mathbf{p} - \mathbf{r}) \cdot (\mathbf{p} - \mathbf{r}) \\ &= -2 \frac{\partial \mathbf{r}}{\partial q_k} \cdot (\mathbf{p} - \mathbf{r}) \,. \end{aligned} \tag{A.24}$$

Looking at the last part of this by component provides

$$\frac{\partial r_i}{\partial q_k} = \frac{\partial}{\partial q_k} (N_{il}(x_l + q_l)) = N_{il} \frac{\partial q_l}{\partial q_k} = N_{ik}.$$

This can be interpreted as the k-th column of the matrix $\mathbf{N}$, which in Matlab's notation would be `N(:,k)`, or

$$\frac{\partial \mathbf{r}}{\partial \mathbf{q}} = \mathbf{N}. \tag{A.25}$$

Next, consider $\frac{\partial d_j^2}{\partial \xi}$ and $\frac{\partial d_j^2}{\partial \eta}$ of equation (3.16).

$$\begin{aligned} \frac{\partial d_j^2}{\partial \xi} &= \frac{\partial}{\partial \xi} (\mathbf{p} - \mathbf{r}) \cdot (\mathbf{p} - \mathbf{r}) \\ &= -2 \frac{\partial \mathbf{r}}{\partial \xi} \cdot (\mathbf{p} - \mathbf{r}) \\ &= -2 \underbrace{(\mathbf{x} + \mathbf{q}) \cdot \mathbf{N}_{,\xi}}_{=\mathbf{r}_{,\xi}} \cdot (\mathbf{p} - \mathbf{r}) \end{aligned} \tag{A.26}$$

From this, one can readily define

$$\mathbf{r}_{,\xi} = \mathbf{N}_{,\xi} (\mathbf{x} + \mathbf{q}) \qquad \mathbf{r}_{,\eta} = \mathbf{N}_{,\eta} (\mathbf{x} + \mathbf{q}) \tag{A.27a}$$

$$\mathbf{r}_{,\xi\xi} = \mathbf{N}_{,\xi\xi} (\mathbf{x} + \mathbf{q}) \qquad \mathbf{r}_{,\eta\eta} = \mathbf{N}_{,\eta\eta} (\mathbf{x} + \mathbf{q}) \tag{A.27b}$$

$$\mathbf{r}_{,\xi\eta} = \mathbf{r}_{,\eta\xi} = \mathbf{N}_{,\xi\eta} (\mathbf{x} + \mathbf{q}) \,. \tag{A.27c}$$

By induction, one can also compute

$$\frac{\partial d_j^2}{\partial \eta} = -2 (\mathbf{x} + \mathbf{q}) \cdot \mathbf{N}_{,\eta} \cdot (\mathbf{p} - \mathbf{r}) = -2 \mathbf{r}_{,\eta} \cdot (\mathbf{p} - \mathbf{r}) \,. \tag{A.28}$$

The term in equation (3.16) $\frac{\partial \boldsymbol{\xi}}{\partial q_k}$ cannot be calculated directly. Here, the author uses implicit function theorem to evaluate the derivatives, and the methods are tailored differently for linear and non-linear elements.

A.4.1.1 Linear Element

The image point $\mathbf{P}$ can also be expressed as

$$\mathbf{P} = [\mathbf{N}(\xi, \eta)(\mathbf{x} + \mathbf{q})]|_{(\xi,\eta)=(\bar{P}_1,\bar{P}_2)} \tag{A.29}$$

where $\bar{\mathbf{P}} = (\bar{P}_1, \bar{P}_2)$ is the inverse mapping of the image point from world coordinates to natural coordinates. By using chain rule, the remaining terms in equation (3.16) can be written as

$$\frac{\partial \xi}{\partial q_k} = \frac{\partial \xi}{\partial \bar{P}_1}\frac{\partial \bar{P}_1}{\partial q_k} + \frac{\partial \xi}{\partial \bar{P}_2}\frac{\partial \bar{P}_2}{\partial q_k} \tag{A.30}$$

$$\frac{\partial \eta}{\partial q_k} = \frac{\partial \eta}{\partial \bar{P}_1}\frac{\partial \bar{P}_1}{\partial q_k} + \frac{\partial \eta}{\partial \bar{P}_2}\frac{\partial \bar{P}_2}{\partial q_k} \tag{A.31}$$

or equivalently,

$$\frac{\partial \boldsymbol{\xi}}{\partial q_k} = \mathbf{J}(\boldsymbol{\xi}, \bar{\mathbf{P}})\frac{\partial \bar{\mathbf{P}}}{\partial q_k} \tag{A.32}$$

where

$$\mathbf{J}(\boldsymbol{\xi}, \bar{\mathbf{P}}) = \begin{bmatrix} \frac{\partial \xi}{\partial \bar{P}_1} & \frac{\partial \xi}{\partial \bar{P}_2} \\ \frac{\partial \eta}{\partial \bar{P}_1} & \frac{\partial \eta}{\partial \bar{P}_2} \end{bmatrix} \tag{A.33}$$

The Jacobian is merely the sensitivity of the closest point on the surface element with respect to the image point both in natural coordinates. It can be easily calculated by using the map shown in Figure A.5.

To calculate the last term in equation (A.32), examine equation (A.29) by component and differentiate

$$P_i = N_{ij}(x_j + q_j) \tag{A.34}$$

with the knowledge of the fact that P is a fixed image point. The total derivative of P with respective to a degree of freedom on FE is zero, while the inverse mapping

has changed.

$$\frac{d}{dq_k}P_i = 0 \tag{A.35}$$

$$= N_{ij}\frac{\partial q_j}{\partial q_k} + \frac{\partial N_{ij}}{\partial q_k}(x_j + q_j) \tag{A.36}$$

$$= N_{ik} + \frac{\partial N_{ij}}{\partial \xi}\frac{\partial \bar{P}_1}{\partial q_k}(x_j + q_j) + \frac{\partial N_{ij}}{\partial \eta}\frac{\partial \bar{P}_2}{\partial q_k}(x_j + q_j) \tag{A.37}$$

again, N_{ik} is the kth column in $\mathbf{N}$. Therefore,

$$\frac{\partial \bar{\mathbf{P}}}{\partial q_k} = -\left[\mathbf{r}_{,\xi}|\mathbf{r}_{,\eta}\right]^{-1}\mathbf{N}(:,k)|_{(\xi,\eta)=(\bar{P}_1,\bar{P}_2)} \tag{A.38}$$

Note that although the inverse mapping exists, the uniqueness can only be guaranteed in the parent element domain. In fact, the determinant of Jacobian of mapping can be less than zero while a point is far away from the element. At such circumstances, one can use partial derivatives of the distance penalty $\frac{\partial d_j^2}{\partial q_k}$ to construct the gradient and Hessian as an approximation.

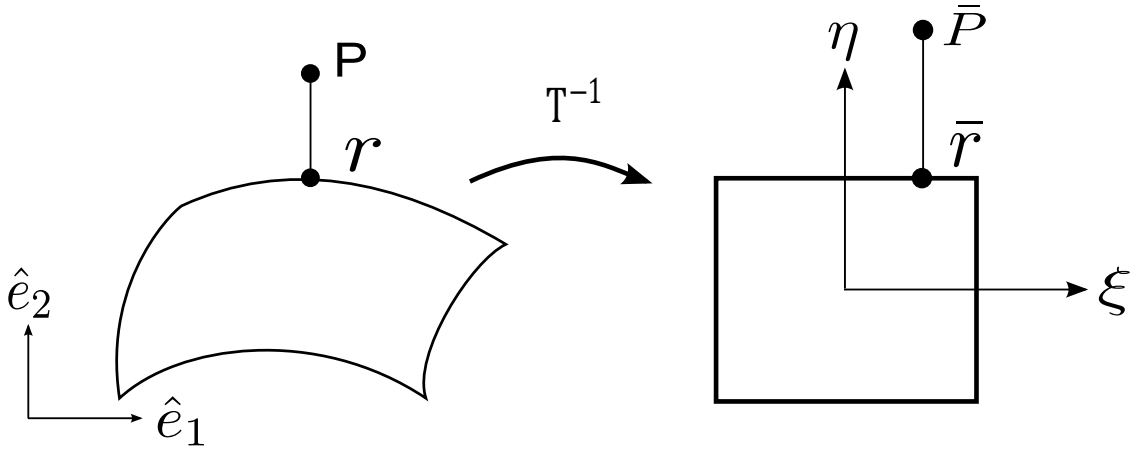


Figure A.4: Illustration of inverse mapping.

A.4.1.2 Higher-order Elements

Let the function

$$\mathbf{g}(\mathbf{q}, \boldsymbol{\xi}) = \frac{\partial d_j^2}{\partial \boldsymbol{\xi}} \tag{A.39}$$

be such that it corresponds to an implicit definition of the minimum distance when

$$\mathbf{g}(\mathbf{q}, \bar{\boldsymbol{\xi}}) = \mathbf{0}\,. \tag{A.40}$$

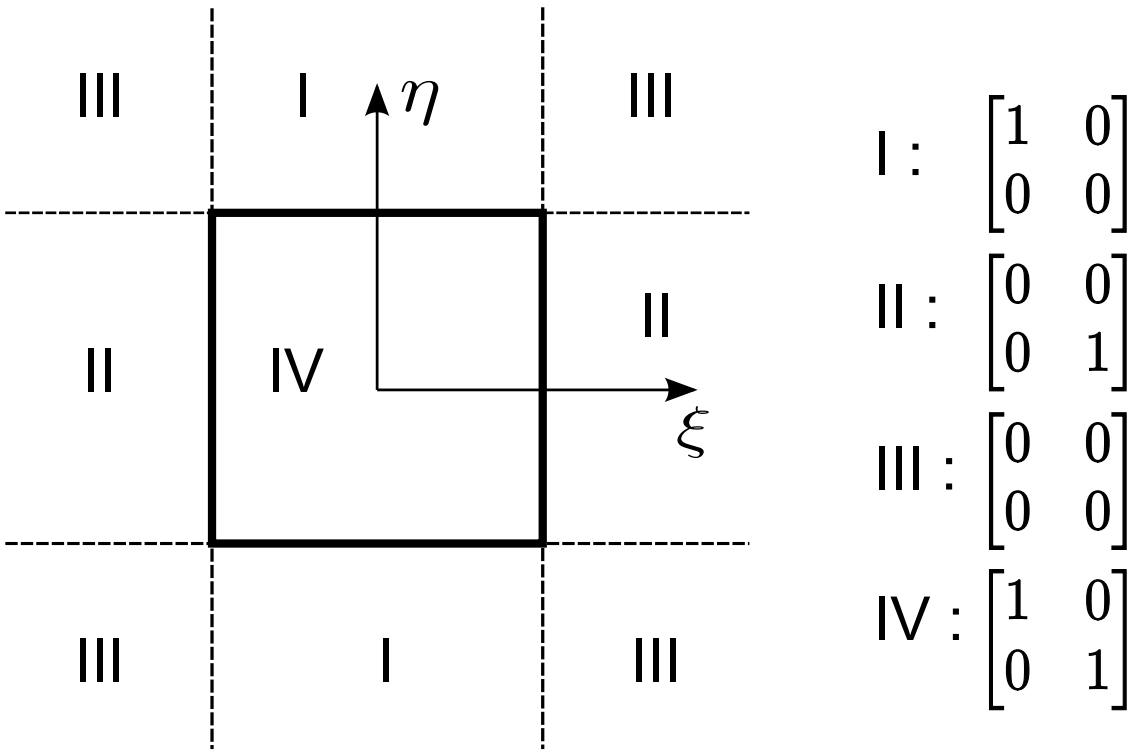


Figure A.5: Jacobian map in equation (A.33). Region IV is the unit square in natural coordinates.

Here, $\bar{\boldsymbol{\xi}} \circ \mathbf{q}$ is the natural coordinates of the point on the surface element corresponding to the solution of equation (3.13) (for the appropriate element). By using the implicit function theorem, one can compute the derivatives of $\boldsymbol{\xi}$ with respect $\mathbf{q}$.

$$\frac{\partial \boldsymbol{\xi}}{\partial \mathbf{q}} = -\left(\frac{\partial \mathbf{g}}{\partial \boldsymbol{\xi}}\right)^{-1} \frac{\partial \mathbf{g}}{\partial \mathbf{q}} \tag{A.41}$$

Working on

$$\frac{\partial \mathbf{g}}{\partial \boldsymbol{\xi}} = \begin{bmatrix} \dfrac{\partial^2 d_j^2}{\partial \xi^2} & \dfrac{\partial^2 d_j^2}{\partial \xi \partial \eta} \\ \dfrac{\partial^2 d_j^2}{\partial \xi \partial \eta} & \dfrac{\partial^2 d_j^2}{\partial \eta^2} \end{bmatrix} \tag{A.42}$$

by piece gives

$$\begin{aligned} \frac{\partial}{\partial \xi}\frac{\partial d_j^2}{\partial \xi} &= -2\frac{\partial}{\partial \xi}\left(\mathbf{r}_{,\xi} \cdot (\mathbf{p} - \mathbf{r})\right) \\ &= -2\left(\mathbf{r}_{,\xi\xi} \cdot (\mathbf{p} - \mathbf{r}) - \mathbf{r}_{,\xi} \cdot \mathbf{r}_{,\xi}\right) \end{aligned} \tag{A.43a}$$

and

$$\frac{\partial}{\partial \eta}\frac{\partial d_j^2}{\partial \eta} = -2\left(\mathbf{r}_{,\eta\eta}\cdot(\mathbf{p}-\mathbf{r}) - \mathbf{r}_{,\eta}\cdot\mathbf{r}_{,\eta}\right). \tag{A.43b}$$

The cross terms are calculated by

$$\begin{aligned}\frac{\partial^2 d_j^2}{\partial \xi \partial \eta} = \frac{\partial^2 d_j^2}{\partial \eta \partial \xi} &= -2\frac{\partial}{\partial \eta}\left(\mathbf{r}_{,\xi}\cdot(\mathbf{p}-\mathbf{r})\right) \\ &= -2\left(\mathbf{r}_{,\xi\eta}\cdot(\mathbf{p}-\mathbf{r}) - \mathbf{r}_{,\eta}\cdot\mathbf{r}_{,\xi}\right).\end{aligned} \tag{A.43c}$$

The last piece of equation (A.42) to formulate is $\frac{\partial \mathbf{g}}{\partial \mathbf{q}}$. Breaking this down by part gives

$$\begin{aligned}\frac{\partial}{\partial \mathbf{q}}\frac{\partial d_j^2}{\partial \xi} &= -2\frac{\partial}{\partial \mathbf{q}}\left(\mathbf{r}_{,\xi}\cdot(\mathbf{p}-\mathbf{r})\right) \\ &= -2\left(\mathbf{r}_{,\xi\mathbf{q}}\cdot(\mathbf{p}-\mathbf{r}) - \mathbf{r}_{,\mathbf{q}}\cdot\mathbf{r}_{,\xi}\right).\end{aligned}$$

Using the definitions above, one can conclude that $\mathbf{r}_{,\xi\mathbf{q}} = \mathbf{N}_{,\xi}$, which simplifies the above expression to

$$\frac{\partial}{\partial \mathbf{q}}\frac{\partial d_j^2}{\partial \xi} = -2\left(\mathbf{N}_{,\xi}\cdot(\mathbf{p}-\mathbf{r}) - \mathbf{N}\cdot\mathbf{r}_{,\xi}\right). \tag{A.44a}$$

Likewise, the other term is computed as

$$\frac{\partial}{\partial \mathbf{q}}\frac{\partial d_j^2}{\partial \eta} = -2\left(\mathbf{N}_{,\eta}\cdot(\mathbf{p}-\mathbf{r}) - \mathbf{N}\cdot\mathbf{r}_{,\eta}\right). \tag{A.44b}$$

The algorithmic steps to compute the gradient is shown in 1.

Algorithm 1 Computing the derivative of the distance penalty from a single voxel.

Precondition: Current state $\mathbf{q}$ of the FEM, pixel point $\mathbf{p}$, $(\bar{\xi}, \bar{\eta})$ solution to (3.13)

1 Compute the derivatives of $\mathbf{r}$ ▷ (A.27)

2 $\left(\dfrac{\partial d_j^2}{\partial \xi}, \dfrac{\partial d_j^2}{\partial \eta}\right)$ ▷ (A.26), (A.28)

3 $\dfrac{\partial \mathbf{g}}{\partial \boldsymbol{\xi}}$ ▷ (A.43)

4 $\dfrac{\partial \mathbf{g}}{\partial \mathbf{q}}$ ▷ (A.44)

5 Solve for $\dfrac{\partial \boldsymbol{\xi}}{\partial \mathbf{q}}$ ▷ (A.41)

6 $\dfrac{\partial d_j^2}{\partial \mathbf{q}} = -2\mathbf{N} \cdot (\mathbf{p} - \mathbf{r})$ ▷ (A.24)

7 $\dfrac{\mathrm{d}}{\mathrm{d}\mathbf{q}} d_j^2 = -2\mathbf{N} \cdot (\mathbf{p} - \mathbf{r}) + \left(\dfrac{\partial d_j^2}{\partial \xi}, \dfrac{\partial d_j^2}{\partial \eta}\right) \cdot \dfrac{\partial \boldsymbol{\xi}}{\partial \mathbf{q}}$ ▷ (3.16)

A.4.2 Gradient of strain energy

Looking at the gradient by element, the derivative can be computed with respect to the local degrees of freedom and then assembled into a global tuple. By using the product rule

$$\frac{\partial U_s^e}{\partial q_k} = \frac{1}{2} \int_{\Omega_o^e} \frac{\partial \boldsymbol{S}}{\partial q_k} : \boldsymbol{E} + \boldsymbol{S} : \frac{\partial \boldsymbol{E}}{\partial q_k} \, \mathrm{d}V_0 \tag{A.45}$$

the problem can be broken up. The tensor $\frac{\partial \boldsymbol{S}}{\partial q_k}$ is computed as part of the usual tangent stiffness matrix and a method to compute it for a Biot material is found in reference [22]. For a Kirchhoff material ($\boldsymbol{S} = \mathfrak{C} : \boldsymbol{E}$), and one can restate equation (A.45) since $\boldsymbol{S} : \boldsymbol{E} = \boldsymbol{E} : \mathfrak{C} : \boldsymbol{E}$ as

$$\begin{aligned} \frac{\partial U_s^e}{\partial q_k} &= \frac{1}{2} \int_{\Omega_o^e} \frac{\partial \boldsymbol{S}}{\partial q_k} : \boldsymbol{E} + \boldsymbol{S} : \frac{\partial \boldsymbol{E}}{\partial q_k} \, \mathrm{d}V_0 \\ &= \frac{1}{2} \int_{\Omega_o^e} \frac{\partial \boldsymbol{E}}{\partial q_k} : \mathfrak{C} : \boldsymbol{E} + \boldsymbol{E} : \mathfrak{C} : \frac{\partial \boldsymbol{E}}{\partial q_k} \, \mathrm{d}V_0. \end{aligned} \tag{A.46}$$

where $\mathfrak{C}$ is the constant fourth-order tensor known as the 2nd elasticity tensor. These derivatives can further be broken down by using the definition of $\boldsymbol{E}$

$$\frac{\partial \boldsymbol{E}}{\partial q_k} = \frac{\partial}{\partial q_k} \left(\boldsymbol{F}^T \boldsymbol{F} - \boldsymbol{I}\right) = \frac{1}{2} \left(\frac{\partial \boldsymbol{F}^T}{\partial q_k} \boldsymbol{F} + \boldsymbol{F}^T \frac{\partial \boldsymbol{F}}{\partial q_k}\right). \tag{A.47}$$

Employing the usual finite-element description, the derivative of $\mathbf{F}$ is computed by component as

$$\frac{\partial F_{ij}}{\partial q_k} = \frac{\partial}{\partial q_k}\left(\frac{\partial N_{il}}{\partial x_j} q_l + \delta_{ij}\right) = \frac{\partial N_{il}}{\partial x_j}\left(\frac{\partial q_l}{\partial q_k}\right) = \frac{\partial N_{il}}{\partial x_j}\delta_{lk} = \frac{\partial N_{ik}}{\partial x_j}. \tag{A.48}$$

By observation $\frac{\partial \boldsymbol{E}}{\partial q_k} = \frac{\partial \boldsymbol{E}^T}{\partial q_k}$, one can combine the previous statements to simplify equation (A.46).

$$\frac{\partial U_s^e}{\partial q_k} = \frac{1}{2}\int_{\Omega_o^e} 2\boldsymbol{E} : \mathfrak{C} : \frac{\partial \boldsymbol{E}}{\partial q_k}\, \mathrm{d}V_0 = \int_{\Omega_o^e} \boldsymbol{S} : \frac{\partial \boldsymbol{E}}{\partial q_k}\, \mathrm{d}V_0 \tag{A.49}$$

This expression is identical to the calculation of $\mathbf{f}_{\text{int}}$. However, this is only true for the Kirchhoff material. The use of a Biot material requires equation (A.45) to be used directly.

A.4.3 Gradient of volume

Computing the derivative of this constraint can be done by element. Since $\mathrm{d}V_0$ is not dependent on $\mathbf{q}$ (by definition) the sensitivity of each element volume to local DOF k can be computed from

$$\frac{\partial V^e}{\partial q_k} = \frac{\partial}{\partial q_k}\int_{\Omega_0} \det\boldsymbol{F}\, \mathrm{d}V_0 = \int_{\Omega_0} \frac{\partial \det\boldsymbol{F}}{\partial q_k}\, \mathrm{d}V_0 \tag{A.50}$$

Recall Jacobi's formula which shows how the derivative of a determinant can be calculated [53]

$$\frac{\partial \det\boldsymbol{F}}{\partial q_k} = \det(\boldsymbol{F})\ \mathrm{tr}\left(\boldsymbol{F}^{-1}\frac{\partial \boldsymbol{F}}{\partial q_k}\right). \tag{A.51}$$

where $\frac{\partial \boldsymbol{F}}{\partial q_k}$ is computed from equation (A.48). Once the element local derivatives are calculated they are assembled (superimposed) into the global derivative tuple.

A.5 Proper orthogonal decomposition

A.5.1 Principal component analysis

Consider matrix A that consists of n sets of data in m-dimensional space. $\mathbf{A} \in R^{n\times m}$. Find a unit vector $\boldsymbol{P} \in R^{m\times 1}$ that best represents the m-dimensional data. Equivalently, find the maximal square sum of the projections of each data on $\boldsymbol{P}$.

$$\begin{aligned} &(\mathbf{A}\boldsymbol{P})^T(\mathbf{A}\boldsymbol{P}) \\ =&\boldsymbol{P}^T\mathbf{A}^T\mathbf{A}\boldsymbol{P} \\ =&\boldsymbol{P}^T\mathbf{V}\boldsymbol{P} \end{aligned} \tag{A.52}$$

Making use of the Lagrange multiplier, construct the Lagrange function

$$L(\boldsymbol{P}) = \boldsymbol{P}^T\mathbf{V}\boldsymbol{P} - \lambda(\boldsymbol{P}^T\boldsymbol{P} - 1) \tag{A.53}$$

and solve

$$\begin{aligned} \frac{\partial L}{\partial \boldsymbol{P}} &= 2\mathbf{V}\boldsymbol{P} - 2\lambda\boldsymbol{P} = \mathbf{0} \\ \Rightarrow &\mathbf{V}\boldsymbol{P} = \lambda\boldsymbol{P} \end{aligned} \tag{A.54}$$

which is an eigenvalue problem. So far, it is concluded that, for a matrix $\mathbf{A}$ consists of sets of data point, the principal components in the sense of least square are the eigenvectors of matrix $\mathbf{A^TA}$.

A.5.2 Singular value decomposition

Singular value decomposition allows that any rectangular matrix to be decomposed as

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^T \tag{A.55}$$

Suppose that $n \geq m$, then

$$\begin{aligned} \mathbf{A}^T\mathbf{A} &= \left(\mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^T\right)^T\left(\mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^T\right) \\ &= \mathbf{V}\boldsymbol{\Lambda}\mathbf{U}^T\mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^T \\ &= \mathbf{V}\boldsymbol{\Lambda}^2\mathbf{V}^T \end{aligned} \tag{A.56}$$

This can be easily related to principal component analysis by post-multiplying equation (A.56) by $\mathbf{V}$; that is

$$\begin{aligned} \mathbf{A}^T\mathbf{A}\mathbf{V} &= \mathbf{V}\boldsymbol{\Lambda}^2\mathbf{V}^T\mathbf{V} \\ &= \mathbf{V}\boldsymbol{\Lambda}^2 \end{aligned} \tag{A.57}$$

The columns of $\mathbf{V}$ are the eigenvectors and the diagonal components of $\mathbf{\Lambda}$ are the square root of eigenvalues. Similarly,

$$\begin{aligned}\mathbf{A}\mathbf{A}^T &= \left(\mathbf{U}\mathbf{\Lambda}\mathbf{V}^T\right)\left(\mathbf{U}\mathbf{\Lambda}\mathbf{V}^T\right)^T \\ &= \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T\mathbf{V}\mathbf{\Lambda}\mathbf{U}^T \\ &= \mathbf{U}\mathbf{\Lambda}^2\mathbf{U}^T\end{aligned} \tag{A.58}$$

Similarly, it can be stated that the columns of the **$\mathbf{U}$ are the principal components of $\mathbf{A}^T$.** Another thing worth noting is the maximum number of non-zero singular values $\mathbf{A}^T$ can have is m. First, consider the eigenvalue of $\mathbf{A}^T\mathbf{A}$, which satisfies

$$\mathbf{A}^T\mathbf{A}\boldsymbol{w} = \gamma\boldsymbol{w} \tag{A.59}$$

Multiplying $\mathbf{A}$ on both sides

$$\left(\mathbf{A}\mathbf{A}^T\right)\mathbf{A}\boldsymbol{w} = \gamma\mathbf{A}\boldsymbol{w} \tag{A.60}$$

which also represents the eigenvalue problem of $\mathbf{A}\mathbf{A}^T$, with the same eigenvalues of $\mathbf{A}^T\mathbf{A}$. Now, let $\boldsymbol{w}'$ be an arbitrary n-m eigenvectors that is orthonormal to other m eigenvectors of $\mathbf{A}\mathbf{A}^T$.

$$\mathbf{U}\boldsymbol{w}' = \mathbf{0} \tag{A.61}$$

After substituting back to the eigenvalue problem of $\mathbf{A}\mathbf{A}^T$

$$\begin{aligned}\left(\mathbf{A}\mathbf{A}^T\right)\mathbf{A}\boldsymbol{w} &= \gamma'\mathbf{A}\boldsymbol{w} \\ \implies \left(\mathbf{A}\mathbf{A}^T\right)\boldsymbol{w}' &= \gamma'\boldsymbol{w}'\end{aligned} \tag{A.62}$$

A.5.3 Implementation

Construct the measurement matrix, which the columns are the displacement of each degree of freedom, and the rows are the snapshots at each time step.

$$\begin{aligned}\mathbf{A}_{dof\times n} &= \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T \\ &= \begin{bmatrix}\mathbf{q}(t_1) & \mathbf{q}(t_2) & \cdots & \mathbf{q}(t_n)\end{bmatrix} \\ &= \begin{bmatrix}\boldsymbol{\phi}_1 & \boldsymbol{\phi}_2 & \cdots \boldsymbol{\phi}_{dof}\end{bmatrix} \left[\begin{array}{ccc} \ddots & & \\ & \lambda_i & \\ & & \ddots \\ \hline & 0 & \end{array}\right] \begin{bmatrix}\mathbf{V}_1^T \\ \mathbf{V}_2^T \\ \vdots \\ \mathbf{V}_n^T\end{bmatrix} \\ &= \begin{bmatrix}\boldsymbol{\phi}_1 & \boldsymbol{\phi}_2 & \cdots \boldsymbol{\phi}_{dof}\end{bmatrix} \begin{bmatrix}\lambda_1\mathbf{V}_1^T \\ \lambda_2\mathbf{V}_2^T \\ \vdots \\ \lambda_n\mathbf{V}_n^T\end{bmatrix}\end{aligned} \tag{A.63}$$

if $dof > n$. Here, $\mathbf{U} \in R^{dof \times dof}$ consists of the column vectors that best represent the space information. Each vector is in $dof \times 1$ space, and it becomes the dominant mode shape of motion if it is mapped back to a 3 dimensional space. Similarly, columns of $\mathbf{V} \in R^{n \times n}$ contain the information of the amplitude change of overall motion through time, where $\mathbf{\Lambda}$ is the amplitude of each mode.

A.6 Fast Fourier transform and its inverse transform

Consider a time signal $f(t)$ and its Fourier Transform pair $F(j\omega)$. In practice, one often only has discrete and finite time measurements. Suppose there are N impulse sampling with interval T

$$\begin{aligned} F(j\omega) &= \int_{-\infty}^{\infty} f(t)e^{-j\omega t}dt \\ &= \int_{0}^{T(N-1)} f(t)e^{-j\omega t}dt \\ &= \sum_{k=0}^{N-1} f[k]e^{-j\omega KT} \end{aligned} \tag{A.64}$$

The fundamental frequency of this finite time signal is obtained by considering the total length of the signal as one cycle.

$$\omega_0 = \frac{1}{NT}\ Hz\ =\ \frac{2\pi}{NT}\ \frac{rad}{s} \tag{A.65}$$

; that is

$$\omega = 0,\ \frac{2\pi}{NT},\ \frac{4\pi}{NT},\ \cdots \frac{2(N-1)\pi}{NT} \tag{A.66}$$

The form of discrete Fourier transform (DFT) can be written as

$$\begin{aligned} F[n] &= \sum_{k=0}^{N-1} f[k]e^{-jn\frac{2\pi}{NT}KT} \\ &= \sum_{k=0}^{N-1} f[k]e^{-j2\pi\frac{nk}{N}} \end{aligned} \tag{A.67}$$

$n = 0,\ 1,\ \cdots N-1$. The inverse DFT is

$$f[k] = \frac{1}{N}\sum_{k=0}^{N-1} F[n]e^{j2\pi\frac{nk}{N}} \tag{A.68}$$

Consider the contribution to $f[k]$ from $F[n]$ and $F[N-n]$

$$f_n[k] = \frac{1}{N}\left\{F[n]e^{j2\pi\frac{nk}{N}} + F[N-n]e^{j2\pi\frac{(N-n)k}{N}}\right\} \tag{A.69}$$

For real $f[k]$,

$$\begin{aligned} F[N-n] =& \sum_{k=0}^{N-1} f[k]e^{-j2\pi\frac{(N-n)k}{N}} \\ =& \sum_{k=0}^{N-1} f[k]e^{j2\pi\frac{nk}{N}}e^{-j2\pi k} \\ =& F^*[n] \end{aligned} \tag{A.70}$$

Therefore,

$$f_n[k] = \frac{1}{N}\left(F[n]e^{j2\pi\frac{nk}{N}} + F^*[n]e^{-j2\pi\frac{nk}{N}}\right) \tag{A.71}$$

Let $F[n] = R_n + jI_n$,

$$\begin{aligned} f_n[k] =& \frac{1}{N}\left\{(R_n + jI_n)\left(\cos\frac{2\pi}{N}nk + j\sin\frac{2\pi}{N}nk\right) + (R_n - jI_n)\left(\cos\frac{2\pi}{N}nk - j\sin\frac{2\pi}{N}nk\right)\right\} \\ =& \frac{1}{N}\left\{2R_n\cos\frac{2\pi}{N}nk - 2I_n\sin\frac{2\pi}{N}nk\right\} \\ =& \frac{2}{N}\|F[n]\|\cos\left(\frac{2\pi}{N}nk + ang(F[n])\right) \end{aligned} \tag{A.72}$$

Appendix B: Representative codes

B.1 Area similarity

The area similarity term is computed with this MATLAB function. x, y, z, and qi represent the reference configuration and the displacement, respectively. ID and IEN define the connectivity of the node number to global tuple and local node number to global node number. Im and Im2 are the measured image and the projections from the deformed FE body, where the projection is defined by K, Extsic, distort. This function is used to discretize the FE body and project it to each image. The percentage area difference is then calculated.

```
function [Area,Im2] = AreaBased_DPenalty_vec(x,y,z,qi,ID,...
    IEN_ws,nel_ws,...
    Im2,Im,...
    quad_rules,nen_list,eltype,...
    SizeIm,K,Extsic,distort,Area0,resol_nel)
% im numbering rule:
% 1: XY, 2: XZ, 3: YZ
%% outsiders count
outside = 0;
 DISCRETE_PTS = cell(1,nel_ws);

for e=1:nel_ws
    %% Get the mesh of the surroundind nodes
    a = 1:9;
    A = IEN_ws(a,e);
    %xe = [x(A); y(A); z(A)];
    xe=x(A);
    ye=y(A);
    ze=z(A);
    qex = qi(ID(1,A));
    qey = qi(ID(2,A));
    qez = qi(ID(3,A));
    lengths=get_element_length(e,x,y,z,qi,eltype,IEN_ws,ID,quad_rules,nen_list);
    resol = resol_nel(e);
    % arc length = "lengths" pixel points
    xi_r = linspace(-1,1,ceil(lengths(1)/resol));
    eta_r = linspace(-1,1,ceil(lengths(2)/resol));
```

```
    [xi_grid,eta_grid] = meshgrid(xi_r,eta_r);
    xi_grid = xi_grid(:);
    eta_grid = eta_grid(:);

    [NN] = el10_ShapeFunctions(xi_grid,eta_grid); % N X # of nodes

    %% Compute the position of the surface point
    pts= [NN*(xe+qex), NN*(ye+qey), NN*(ze+qez)]';%3XN
     DISCRETE_PTS{e} = pts;
end
  pts = cell2mat(DISCRETE_PTS);
  clear DISCRETE_PTS

    LEN = length(pts(1,:));
    for cam = 1:3
        xp=make_Npts_reprojection(pts,K(:,:,cam),Extsic(:,:,cam),distort(:,cam),LEN);
        xp=round(xp);
        LEN2 = length(xp(1,:));
        xp = is_in_Im_pts(xp,SizeIm);
        if isempty(xp)
            outside = outside + LEN2;
        else
            LEN3 = length(xp(1,:));
            out = LEN2-LEN3;
            outside = outside + out;
            idx = sub2ind(SizeIm,xp(2,:),xp(1,:),cam*ones(1,LEN3));
            Im2(idx)=true;
        end

    end


%% sum up the area difference
Area = zeros(3,1);
for cam = 1:3
    Intersec = Im2(:,:,cam).* Im(:,:,cam);
    Union    = Im2(:,:,cam) | Im(:,:,cam);
    ShapeDiff = Union - Intersec;
    Area(cam) = sum(ShapeDiff(:))/Area0(cam);
end
Area = sum(Area)+outside/sum(Area0);
Area = Area*100/3;% mean percentage difference
end
```

B.2 Assembly for the gradient of similarity penalty

B.2.1 Finite difference scheme for area similarity

```
%% nodal gradient:
% prepare the length of each element
len_e = zeros(nel_ws,2);
for i1 = 1:nel_ws
    lengths =get_element_length(i1,x,y,z,...
        qi,eltype_ws,IEN_ws,ID,quad_rules,nen);
    len_e(i1,:) = lengths;
end

% moves nodes on surface element only
NodeIdx_ws = unique(IEN_ws(:));
if mod(optimize.iteration2,10)~=0
    NodeIdx_ws = intersect(NodeIdx_ws,activeD_idx);
end

nnp_ws = length(NodeIdx_ws);
parQI = WorkerObjWrapper(qi);
dD = zeros(nnp*3,1);
GradSeg = zeros(3,nnp_ws);
parfor i = 1:nnp_ws
    j1 = NodeIdx_ws(i);
    GradSeg(:,i) = nodal_Area_gradient_mex(j1,...
    parX.Value,parY.Value,parZ.Value,parQI.Value,ID,...
    IEN_ws,nel_ws,...
    parSpatBIm.Value,parSpatIm.Value,...
    quad.nt,quad.w,quad.xi,...
    SizeIm,cam.K,cam.Extsic,cam.distort,Area0,resol_nel,resol_dof,len_e);
end
for i1 = 1:nnp_ws
    j1 = NodeIdx_ws(i1);
    k1 = ID(:,j1);
    dD(k1) = dD(k1) + GradSeg(:,i1);
end
```

```
function [grad] = nodal_Area_gradient(nodeIdx,...
    x,y,z,qi,ID,...
    IEN_ws,nel_ws,...
    Im2,Im,...
    quad_nt,quad_w,quad_xi,...
    SizeIm,K,Extsic,distort,Area0,resol_nel,resol_dof,len_e)%#codegen
%% initialize
grad = zeros(3,1);
```

```
outside = 0;
%% find the element(s) that contains the node
[local_node,ele_num] = find(IEN_ws==nodeIdx);
len_element = length(ele_num);
global_ele_idx = setdiff(1:nel_ws,ele_num);

%% create the shared Im for later perturbation
for e1=1: (nel_ws-len_element)
    e = global_ele_idx(e1);
    %% Get the mesh of the surroundind nodes
    [Im2,outside] = element_projection(Im2,e,...
           x,y,z,IEN_ws,qi,ID,...
           quad_nt, quad_xi, quad_w,resol_nel,...
           K,Extsic,distort,SizeIm,outside,len_e(e,:));
end
%% now perturb
for dof = 1 : 3
    Im_dx1 = Im2;
    Im_dx2 = Im2;
    outside_dx1 = outside;
    outside_dx2 = outside;
    global_dof = ID(dof,nodeIdx);
    dx = resol_dof(global_dof);
    qi1 = qi;qi2 = qi;
    qi1(global_dof) = qi(global_dof)+dx;
    qi2(global_dof) = qi(global_dof)-dx;
    % make the projection of the elements relavent to that perturb point
    for e1 = 1:len_element
        e = ele_num(e1);
        [Im_dx1,outside_dx1] = element_projection(Im_dx1,e,...
            x,y,z,IEN_ws,qi1,ID,...
            quad_nt, quad_xi, quad_w,resol_nel,...
            K,Extsic,distort,SizeIm,outside_dx1,len_e(e,:));
        [Im_dx2,outside_dx2] = element_projection(Im_dx2,e,...
            x,y,z,IEN_ws,qi2,ID,...
            quad_nt, quad_xi, quad_w,resol_nel,...
            K,Extsic,distort,SizeIm,outside_dx2,len_e(e,:));
    end
    % calculate gradient
    % second order f' = (f_{i+1}-f_{i-1})/(2dx) + O(dx^2)
    f1 = get_percentage_area_diff(Im,Im_dx1,Area0,outside_dx1);
    f2 = get_percentage_area_diff(Im,Im_dx2,Area0,outside_dx2);
    dA = 0.5*(f1-f2)/dx;
    grad(dof) = dA;
end
end
```

```
function [Im2,outside] = element_projection(Im2,e,...
          x,y,z,IEN_ws,qi,ID,...
          quad_nt, quad_xi, quad_w,resol_nel,...
          K,Extsic,distort,SizeIm,outside,len_e)
a = 1:9;
A = IEN_ws(a,e);
%xe = [x(A); y(A); z(A)];
xe=x(A);
ye=y(A);
ze=z(A);
qex = qi(ID(1,A));
qey = qi(ID(2,A));
qez = qi(ID(3,A));
% lengths = el10_lengths( xe, ye, ze, qex, qey, qez, ...
%    quad_nt, quad_xi, quad_w);
resol = resol_nel(e);
% arc length = "lengths" pixel points
xi_r = linspace(-1,1,ceil(len_e(1)/resol));
eta_r = linspace(-1,1,ceil(len_e(2)/resol));


for i1 = 1:length(xi_r)
    xi = xi_r(i1);
    for i2 = 1:length(eta_r)
        eta = eta_r(i2);
        %           len_temp=len_temp+1;
        [NN] = el10_ShapeFunctions(xi,eta);

        %% Compute the position of the surface point
        pts= [NN*(xe+qex);
            NN*(ye+qey);
            NN*(ze+qez)];
        for cam = 1:3
            xp=make_reprojection(pts,K(:,:,cam),Extsic(:,:,cam),distort(:,cam));
            xp=round(xp);
            flag = is_in_Im(xp,SizeIm);
            if flag
                Im2(xp(2),xp(1),cam)=true;
            else outside=outside+1;
            end
        end
    end
end

end
```

```
function [Area]=get_percentage_area_diff(Im,Im2,Area0,outside)
Area = zeros(3,1);
for cam = 1:3
    Intersec = Im2(:,:,cam).* Im(:,:,cam);
    Union    = Im2(:,:,cam) | Im(:,:,cam);
    ShapeDiff = Union - Intersec;
    Area(cam) = sum(ShapeDiff(:))/Area0(cam);
end
Area = sum(Area)+outside/sum(Area0);
Area = Area*100/3;

end
```

B.2.2 Gradient of the distance penalty of one element

```
function [dD_temp1,dD_temp2,ddD_temp1,ddD_row,ddD_col]=...
    derivatives_of_D2(P,IEN,ID,IM,pos,e,pos_nat, ...
    x,y,z,qi) %#codegen

% assemble the gradient of distance penalty element-wise
% P : location of image points that are registered to the e-th element
% pos : the location of the point on the FEM body registering the point P
% pos_nat : the corresponding coordinate in natural coordinate.

% example:
% condider one image point P(:,i)
% the location of the closest point on the surface is pos(:,i)
% pos(:,i) can also be calculated by N_jX_j evaluated at
% (xi,eta) = pos_nat
% IM : 1: XY image, 2: YZ image , 3: XZ image
nen_e=9;

LEN=length(P(1,:));
%% temp
dD_temp1=zeros(27,1); % value
dD_temp2=zeros(27,1); % location
ddD_temp1=zeros(27,27);
ddD_row=zeros(27,27);
ddD_col=zeros(27,27);
% dD_temp1 : save the value of gradient of Distance penalty
%           [dx1 dy1 dz1 dx2 dy2 dz2 dx3....] in local node sequence
% dD_temp2 : global equation number = dD_temp2(local equatio number)
%           gradient(dD_temp2)=dD_temp1;

%% calculate the value and save the location
```

```
if LEN~=0
    A= IEN(:,e);
    switch IM
        case 1
            nodes=[x(A)+qi(ID(1,A)) y(A)+qi(ID(2,A)) ];
        case 2
            nodes=[ y(A)+qi(ID(2,A)) z(A)+qi(ID(3,A))];
        case 3
            nodes=[x(A)+qi(ID(1,A)) z(A)+qi(ID(3,A)) ];
        otherwise
            nodes=zeros(9,2);
    end
    %% loop over image points
    for m=1:LEN
        dD_temp=zeros(2,1);
        p1=P(:,m); % image point
        pos1=pos(:,m); % corresponding closest point
        pos_nat1 = pos_nat(:,m);


        [N ,Nr ,Ns]= el10_ShapeFunctions(pos_nat1(1),pos_nat1(2));
        [Nrr,Nss,Nrs]= el10_ShapeFunctionsD2(pos_nat1(1),pos_nat1(2));
        %-------------------------------------------------------------------------
        %   calculate the A matrix in the implicit function
        %-------------------------------------------------------------------------
        A = zeros(2);
        A(1,1) = Nrr*nodes*(p1-pos1)-( Nr*nodes(:,1) )^2-( Nr*nodes(:,2)
            )^2;
        A(1,2) = Nrs*nodes*(p1-pos1)-( Nr*nodes(:,1) )*( Ns*nodes(:,1)
            )...
            -( Nr*nodes(:,2) )*( Ns*nodes(:,2) );
        A(2,1) = Nrs*nodes*(p1-pos1)-( Nr*nodes(:,1) )*( Ns*nodes(:,1)
            )...
            -( Nr*nodes(:,2) )*( Ns*nodes(:,2) );
        A(2,2) = Nss*nodes*(p1-pos1)-( Ns*nodes(:,1) )^2-( Ns*nodes(:,2)
            )^2;

        for i=1:nen_e % loop over the local nodes
            % local node registering
            switch IM
                case 1
                    idx=[(i-1)*3+1 (i-1)*3+2]';
                case 2
                    idx=[(i-1)*3+2 (i-1)*3+3]';
                case 3
                    idx=[(i-1)*3+1 (i-1)*3+3]';
                otherwise % put this to complete the cases
```

```
                idx=[0 0]';
            end
            %-------------------------------------------------------------------------
            %  solve the implicit function
            %-------------------------------------------------------------------------
            Bx = [-Nr(i);-Ns(i)]*(p1(1)-pos1(1)) +
                N(i)*[Nr*nodes(:,1);Ns*nodes(:,1)];
            By = [-Nr(i);-Ns(i)]*(p1(2)-pos1(2)) +
                N(i)*[Nr*nodes(:,2);Ns*nodes(:,2)];
            drdxi = A\Bx;
            drdyi = A\By;
            drdx = drdxi(1);
            dsdx = drdxi(2);
            drdy = drdyi(1);
            dsdy = drdyi(2);
            if abs( pos_nat1(1) )==1 % r=+-1
                drdx=0;
                drdy=0;
            end
            if abs( pos_nat1(2) )==1 % s=+-1
                dsdx=0;
                dsdy=0;
            end

            dNdxi = Nr*drdx+Ns*dsdx;
            dNdyi = Nr*drdy+Ns*dsdy;
            %-------------------------------------------------------------------------
            %  calculate [\partial D \partial xj ;\partial D \partial yj ]
            %-------------------------------------------------------------------------

                dD_temp(1) = 2*(p1(1)-pos1(1))*(-N(i)-dNdxi*nodes(:,1) )...
                    +2*(p1(2)-pos1(2))*(   -dNdxi*nodes(:,2) );
                dD_temp(2) = 2*(p1(1)-pos1(1))*( -dNdyi*nodes(:,1) )...
                    +2*(p1(2)-pos1(2))*(-N(i)-dNdyi*nodes(:,2) );
%               dD_temp = 2*(p1-pos1)*(-N(i));% partail derivatives
                % dD_temp: [\partial D \partial xj ;\partial D \partial yj
                    ]
                dD_temp1(idx)=dD_temp1(idx)+dD_temp;


        end
    end
    % save the location
    for i=1:nen_e
        j=IEN(i,e); % global node number
        idx=ID(:,j);% equation number
        dD_temp2((i-1)*3+1:(i-1)*3+3)=idx;
```

```
end
%% hessian matrix : partial derivative
% assemble Hessian
for m=1:LEN
    N = el10_ShapeFunctions(pos_nat(1,m),pos_nat(2,m));
    for i=1:nen_e
        for j=1:nen_e
            temp=2*N(i)*N(j);

            switch IM
                case 1
                    %-----------------------------------------------------------------
                    %    original assembly process
                    %
                        ddD(ID(1,Ai),ID(1,Aj))=ddD(ID(1,Ai),ID(1,Aj))+temp;
                    %
                        ddD(ID(2,Ai),ID(2,Aj))=ddD(ID(2,Ai),ID(2,Aj))+temp;
                    %-----------------------------------------------------------------
                    row1= (i-1)*3+1;
                    col1= (j-1)*3+1;
                    row2= (i-1)*3+2;
                    col2= (j-1)*3+2;
                    ddD_temp1(row1,col1)=ddD_temp1(row1,col1)+temp;
                    ddD_temp1(row2,col2)=ddD_temp1(row2,col2)+temp;
                case 2
                    row1= (i-1)*3+2;
                    col1= (j-1)*3+2;
                    row2= (i-1)*3+3;
                    col2= (j-1)*3+3;
                    ddD_temp1(row1,col1)=ddD_temp1(row1,col1)+temp;
                    ddD_temp1(row2,col2)=ddD_temp1(row2,col2)+temp;
                case 3
                    row1= (i-1)*3+1;
                    col1= (j-1)*3+1;
                    row2= (i-1)*3+3;
                    col2= (j-1)*3+3;
                    ddD_temp1(row1,col1)=ddD_temp1(row1,col1)+temp;
                    ddD_temp1(row2,col2)=ddD_temp1(row2,col2)+temp;
            end
        end
    end
end
% save the location
% i,j  ith element x => jx, jy, jz, (j+1)x, (j+1)y, (j+1)z,(j+2)x,
    (j+2)y, (j+2)z
%      ith element y => jx, jy, jz, (j+1)x, (j+1)y, (j+1)z,(j+2)x,
    (j+2)y, (j+2)z
```

```
    % or in other words , partial D /partial--
%      x1^2  x1y1 x1z1 x1x2 x1y2 x1z2 ...
%      y1x1  y1^2 y1z1 y1x2 y1y2 y1z2 ...
%      z1x1  z1y1 z1^2 z1x2 z1y2 z1z2 ...


%  ddD_row:
%  x1 x1 x1
%  y1 y1 y1
%  z1 z1 z1

%  ddD_col
%  x1 y1 z1
%  x1 y1 z1
%  x1 y1 z1

    for i=1:nen_e
        Ai=IEN(i,e);
        row=ID(:,Ai); % global equation number of the rows
        for j=1:nen_e
            % partial derivatives of D w.r.t. i and j element
            Aj=IEN(j,e);
            col=ID(:,Aj)'; % global equation number of the cols
            ddD_row((i-1)*3+1:(i-1)*3+3,(j-1)*3+1:(j-1)*3+3)= ...
                [row row row];
            ddD_col((i-1)*3+1:(i-1)*3+3,(j-1)*3+1:(j-1)*3+3)= ...
                [col;col;col];
        end
    end
end
```

Bibliography

[1] T. W. Sims. *A Structural Dynamic Analysis of a Manduca Sexta Forewing*. PhD dissertation, Air Force Institute of Technology, March 2010.

[2] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[3] T. L. Hedrick. Software techniques for two- and three-dimensional kinematic measurements of biological and biomimetic systems. *Bioinspiration & Biomimetics*, 3:034001, 2008. doi:10.1088/1748-3182/3/3/034001.

[4] V. M. Ortega-Jimenez, J. S. Greeter, R. Mittal, and T. L. Hedrick. Hawkmoth flight stability in turbulent vortex streets. *Journal of Experimental Biology*, 216 (24):4567–4579, 2013. doi:10.1242/eb.089672.

[5] S. Dong. BDF-like methods for nonlinear dynamic analysis. *Journal of Computational Physics*, 229(8):3019–3045, 2010. ISSN 0021-9991. doi:10.1016/j.jcp.2009.12.028.

[6] C. Koehler, Z. Liang, Z. Gaston, H. Wan, and H. Dong. 3d reconstruction and analysis of wing deformation in free-flying dragonflies. *The Journal of experimental biology*, 215(17):3018–3027, 2012.

[7] J. Song, H. Luo, and T. L. Hedrick. Three-dimensional flow and lift characteristics of a hovering ruby-throated hummingbird. *Journal of The Royal Society Interface*, 11(98):20140541, 2014. doi:10.1098/rsif.2014.0541.

[8] A. P. Willmott and C. P. Ellington. The mechanics of flight in the hawkmoth manduca sexta. *Journal of Experimental Biology*, 200:2705–2722, 1997.

[9] B. W. Tobalske, D. R. Warrick, C. J. Clark, D. R. Powers, T. L. Hedrick, G. A. Hyder, and A. A. Biewener. Three-dimensional kinematics of hummingbird flight. *Journal of Experimental Biology*, 210(13):2368–2382, 2007.

[10] E. I. Fontaine, F. Zabala, M. H. Dickinson, and J. W. Burdick. Wing and body motion during flight initiation in drosophila revealed by automated visual tracking. *Journal of Experimental Biology*, 212:1307–1323, 2009. doi:10.1242/jeb.025379.

[11] A. P. Willmott and C. P. Ellington. The mechanics of flight in the hawkmoth manduca sexta. ii. aerodynamic consequences of kinematic and morphological variation. *The Journal of experimental biology*, 200(21):2723–2745, 1997.

[12] R. Ramamurti and W. C. Sandberg. A three-dimensional computational study of the aerodynamic mechanisms of insect flight. *Journal of Experimental Biology*, 205:1507–1518, 2002.

[13] J. Song, H. Luo, and T. L. Hedrick. Performance of a quasi-steady model for hovering hummingbirds. *Theoretical and Applied Mechanics Letters*, 5(1): 50–53, 2015.

[14] B. A. Roccia, S. Preidikman, J. C. Massa, and D. T. Mook. Modified unsteady vortex-lattice method to study flapping wings in hover flight. *AIAA journal*, 51(11):2628–2642, 2013.

[15] H. Wan, H. Dong, and K. Gai. Computational investigation of cicada aerodynamics in forward flight. *Journal of The Royal Society Interface*, 12(102): 20141116, 2015.

[16] H. Dong and Z. Liang. Effects of ipsilateral wing-wing interactions on aerodynamic performance of flapping wings. In *Proceedings of 48th AIAA aerospace sciences meeting including the new horizons forum and aerospace exposition*, pages 2010–71, 2010.

[17] M. H. Dickinson, F.-O. Lehmann, and S. P. Sane. Wing rotation and the aerodynamic basis of insect flight. *Science*, 284(5422):1954–1960, 1999. doi:10.1126/science.284.5422.1954.

[18] B. Cheng, J. Roll, Y. Liu, D. R. Troolin, and X. Deng. Three-dimensional vortex wake structure of flapping wings in hovering flight. *Journal of The Royal Society Interface*, 11(91):20130984, 2014.

[19] M. Vanella, T. Fitzgerald, S. Preidikman, E. Balaras, and B. Balachandran. Influence of flexibility on the aerodynamic performance of a hovering wing. *Journal of Experimental Biology*, 212(1):95–105, 2009. doi:10.1242/jeb.016428.

[20] C.-K. Kang, H. Aono, C. E. Cesnik, and W. Shyy. Effects of flexibility on the aerodynamic performance of flapping wings. *Journal of fluid mechanics*, 689: 32–74, 2011.

[21] L. A. Miller and C. S. Peskin. Flexible clap and fling in tiny insect flight. *Journal of Experimental Biology*, 212(19):3076–3090, 2009.

[22] T. Fitzgerald. *Nonlinear fluid-structure interactions in flapping wing systems*. PhD dissertation, University of Maryland College Park, 2013. `http://hdl.handle.net/1903/14825`.

[23] T. Nakata and H. Liu. Aerodynamic performance of a hovering hawkmoth with flexible wings: a computational approach. *Proceedings of the Royal Society of London B: Biological Sciences*, 279(1729):722–731, 2012.

[24] S. Ramananarivo, R. Godoy-Diana, and B. Thiria. Rather than resonance, flapping wing flyers may play on aerodynamics to improve performance. *Proceedings of the National Academy of Sciences*, 108(15):5964–5969, 2011.

[25] L. Ristroph, G. J. Berman, A. J. Bergou, Z. J. Wang, and I. Cohen. Automated hull reconstruction motion tracking (hrmt) applied to sideways maneuvers of free-flying insects. *Journal of Experimental Biology*, 212(9):1324–1335, 2009.

[26] C. Geuzaine and J.-F. Remacle. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331, 2009. doi:10.1002/nme.2579.

[27] A. H. Nayfeh and B. Balachandran. *Applied Nonlinear Dynamics: Analytical, Computational, and Experimental Methods*. Wiley, 1995.

[28] T. Poggio and V. Torre. *Ill-posed problems and regularization analysis in early vision*. Massachusetts Institute of Technology, 1984.

[29] A. N. Tikhonov and V. I. A. Arsenin. *Solutions of ill-posed problems*. Scripta series in mathematics. Winston, 1977. ISBN 9780470991244.

[30] B. K. Horn and B. G. Schunck. Determining optical flow. In *1981 Technical Symposium East*, pages 319–331. International Society for Optics and Photonics, 1981.

[31] D. Terzopoulos. Multilevel computational processes for visual surface reconstruction. *Computer Vision, Graphics, and Image Processing*, 24(1):52–96, 1983. doi:10.1016/0734-189X(83)90020-8.

[32] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International journal of computer vision*, 1(4):321–331, 1988.

[33] D. Terzopoulos. On matching deformable models to images. Technical Report 60, Schlumberger Palo Alto Research, 1987. Reprinted in *Topical Meeting on Machine Vision*, Technical Digest Series, Vol. 12, 1987, pp. 160-167.

[34] R. Szeliski. *Computer vision: algorithms and applications*. Springer, 2010.

[35] D. Terzopoulos. Regularization of inverse visual problems involving discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(4):413–424, 1986. doi:10.1109/tpami.1986.4767807.

[36] D. Terzopoulos, A. Witkin, and M. Kass. Constraints on deformable models: Recovering 3D shape and nonrigid motion. *Artificial Intelligence*, 36(1):91–123, 1988. doi:10.1016/0004-3702(88)90080-X.

[37] L. D. Cohen. On active contour models and balloons. *CVGIP: Image understanding*, 53(2):211–218, 1991.

[38] P. Fua and Y. G. Leclerc. Model driven edge detection. *Machine Vision and Applications*, 3(1):45–56, 1990.

[39] D. Metaxas and D. Terzopoulos. Constrained deformable superquadrics and nonrigid motion tracking. In *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on*, pages 337–343. IEEE, 1991.

[40] Y. Yu, S. Zhang, K. Li, D. Metaxas, and L. Axel. Deformable models with sparsity constraints for cardiac motion analysis. *Medical Image Analysis*, 18 (6):927–937, 2014. doi:10.1016/j.media.2014.03.002.

[41] L. D. Cohen and I. Cohen. Finite-element methods for active contour models and balloons for 2-d and 3-d images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(11):1131–1147, 1993.

[42] M. Ferrant, A. Nabavi, B. Macq, F. Jolesz, R. Kikinis, S. K. Warfield, et al. Registration of 3-d intraoperative mr images of the brain using a finite-element biomechanical model. *Medical Imaging, IEEE Transactions on*, 20(12):1384–1397, 2001.

[43] M. Bucki, C. Lobos, and Y. Payan. A fast and robust patient specific finite element mesh registration technique: application to 60 clinical cases. *Medical Image Analysis*, 14(3):303–317, 2010. doi:10.1016/j.media.2010.02.003.

[44] J. Arora. *Introduction to optimum design*. Academic Press, 2004.

[45] B. G. Baumgart. Geometric modeling for computer vision. Technical report, DTIC Document, 1974.

[46] G. A. Holzapfel. *Nonlinear Solid Mechanics*. Wiley, 2000.

[47] A. C. Antoulas. *Approximation of large-scale dynamical systems*. Society for Industrial and Applied Mathematics, Philadelphia, 2005.

[48] B. Feeny. A complex orthogonal decomposition for wave motion analysis. *Journal of Sound and Vibration*, 310(1):77–90, 2008.

[49] T. Fitzgerald, M. Valdez, and B. Balachandran. A comparison of computational models for fluid-structure interaction studies of flexible flapping wing systems. In *The 49th AIAA Aerospace Sciences Meeting, Orlando*, pages 4–7, 2011.

[50] S. Preidikman. *Numerical Simulations of Interactions Amoung Aerodynamics, Structural Dynamics, and Control Systems*. PhD dissertation, Virginia Polytechnic Institute and State University, 1998.

[51] P. Konstadinopoulos, D. Mook, and A. Nayfeh. A numerical method for general unsteady aerodynamics. *AIAA paper*, pages 81–1877, 1981.

[52] J. Heikkila and O. Silvén. A four-step camera calibration procedure with implicit image correction. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 1106–1112. IEEE, 1997.

[53] J. R. Magnus and H. Neudecker. *Matrix Differential Calculus with Applications in Statistics and Economics*. Wiley, 3 edition, 2007.